2012 Edition

SAS® to Excel: Give Them the Report Exactly the Way That They Want It

LeRoy Bessler PhD, Bessler Consulting and Research
Mequon, Milwaukee, Wisconsin, USA, Le_Roy_Bessler@wi.rr.com

Abstract

To create complex, highly formatted Excel spreadsheets from a SAS program, Dynamic Data Exchange (DDE) is the tool preferred by numerous SAS users, including me. I get requests nearly every week from all over the world for my free DDE toolkit. Though the oldest SAS-to-Excel solution, DDE nevertheless works with Excel 2010 and Excel 2007. Non-DDE solutions can be easy when the problems are easy.

DDE supercharges your use of SAS with Excel. SAS DDE programs can load worksheets, and format them maximally without point-and-click. Almost anything doable directly in Excel is doable with DDE commands from your program. You can access data, even to the level of a specific cell, a row or column, part of a row or column, or a range of contiguous rows or columns, or can work with an entire worksheet. Then you can write, read, or format data. Why would you want to run Excel from SAS? You can create a hands-off production job to be automatically scheduled by the computer to access and analyze the data, load the spreadsheet, and format it. You can even attach the report to, or include a link to it in, a SAS-dispatched email message (which does not require DDE). In any case, if you find yourself preparing the same highly formatted Excel report over and over, why not automate it?

The intended reader for this paper is any level of SAS user who needs to deliver information via Excel.

Introduction

The main body of this paper (exclusive of the Appendices) is my original introduction to a SAS-with-Excel self-study collection of sixty macros and twenty-six sample programs (with comments), and supporting files, which are available at no charge from me via email (at Le_Roy_Bessler@wi.rr.com).

My original paper on this topic, “SAS-with-Excel Application Development: Tools and Techniques”, was constrained by the SUGI 2006 Proceedings publication limit of twenty pages, and by lack of author interest at the time in two interesting problems: working with pivot tables from SAS and getting graphs into Excel. Appendices 1 to 4 are from an Expanded Edition of the paper developed for MidWest SAS Users Group Conference (MWSUG) 2007, where the page count limit was waived. Appendices 5 and 6, on using SAS DDE to create non-Excel file types, was added for MWSUG 2010. Also, Appendix 1 was updated for MWSUG 2010. I regret any inconsistencies in font use across this evolving document.

To find out more about the Excel commands used here to work with DDE, or to find other commands or command options not used here, you must consult the documentation in Macrofun.hlp. To get that information, you must install Macrofun.hlp by initially downloading Macrofun.exe from the Microsoft Download Center, and then run the .exe file. If you are not experienced using the Download Center, you can find the right part of it by using google.com to search for “macrofun.exe”, or instead you could try this URL: http://support.microsoft.com:80/support/kb/articles/Q128/1/85.ASP.

NOTE: I want to answer a question I get about cell merge. Whenever I have run into a limitation on DDE, my solution is to create a template Excel workbook, building in the needed feature, and load that template, doing any further dynamic customization with DDE after the initial load. So, one can manually create a template with merged cells where needed, merge the intended content with SAS DATA step pre-processing, and use DDE to load the target cell (made out of the merged cells). Not an ideal situation, but for the power that DDE gives you as compared to ALL other alternatives, it's OK.

DDE: Dynamic Data Exchange

Client-Server Architecture

Your SAS session is a client. It opens Excel, which acts as a server. Your SAS program:

1. Starts/Executes Excel
2. Establishes a channel to talk to Excel
3. Opens a new/existing XLS file (workbook)
4. Sends commands to Excel
5. Saves, or Saves As (new or different), XLS file
6. Exits/Stops Excel

What Can You Do with DDE?

Anything that you might do in Excel with your keyboard and mouse! Here I cover only a subset of the input/output and formatting possibilities.

When DDE cannot do it (rare), you can pre-record an Excel macro and run it with DDE.

Some of the Cell Formatting Options

- Font controls available include: font, style, size, color, and underline.
- Cell background can be filled with color.
- Cell sizing is possible, and column or row AutoFit and column or row Hide/Unhide.
- Cell alignment, text wrap, etc. can be done.
- Borders can be created, e.g., to section the worksheet.
- Cell content can be indented, using two tricks shown in sample programs.

Some of the Worksheet Controls

- Select worksheet, cells, rows, columns
- Freeze/Unfreeze panes
- Turn AutoFilter On/Off
- Generate subtotals in a worksheet
- Control/Change levels for Excel subtotals (or create multiple custom sheets for various levels of totals)
- Activate a specific worksheet in a workbook
- Create additional worksheets in a workbook
- Delete unused worksheets in a workbook

Some of the Other Manipulation Possibilities

- Insert/Delete columns or rows
- Find, Find and Replace cell contents
- Copy cells, Move cells
- Use formulas to create new column(s)
- Use Zoom to shrink or enlarge a worksheet
- Control the Message text box in the lower left corner of the Excel window

The Mechanics of DDE and the Basic SAS-with-Excel Interaction

Channel for DDE Commands

DDE commands require use of special SAS filename statements:

```
filename YourFileRef dde "excel|system";
```

YourFileRef is arbitrary 1-to-8 characters. I like **DDEcmds**.

FILENAME Statement Assignments to Read or Write Data with DDE

Identification of the current selection of worksheet and a rectangle of cells for read or write is accomplished with:

```
EXCEL|YourSheet!RppppCqqq:RssssCttt
```

ppppp is 1-to-5-digit start/top row number
qqq is 1-to-3-digit start/left column number
sssss is end/bottom row number
ttt is end/right column number

Row/column numbers with leading zeros are permissible.

Other ways to make selections of cells are as follows:

RppppCqqq - single cell
Rppppp:Rssssss - range of rows
Rppppp - single row
Cqqq:Cttt - range of columns
Cqqq - single column

See the section “Insert a Heading for the New Column 4” for a simple example of the FILENAME statement used for writing data with DDE. For more examples of writing data to Excel from SAS, see Reference 1.

Coding the Cell Selection

You are not required to permanently hard code it. You can use DDE select functions to operate based on criteria assigned by you (as in some examples below). Or you can use macro program code to determine the selection target dynamically (as in examples from the tools package available via email).

Excel DDE Commands

Microsoft developed DDE commands for an earlier version of Excel than the current one. I have used them with various versions of Excel and Windows. Almost all of the commands (that I have tried) work for me, but I can offer no guarantees.

The commands are documented in Macrofun.hlp, which you can download from Microsoft as mentioned above. "Macrofun" is an abbreviation for "Macro Functions", not "Macro Fun", but these macros functions are fun to use. They do not require use of Excel macros.

Please be aware that not every DDE command works (e.g., rename worksheets, create subtotals, . . .), and some command options may not work. Nevertheless, what *does* work is an enormous tool set!

true / false Values for DDE Commands

These are used to turn options On / Off in DDE command parameter assignments. For some commands you are required to use 1 / 0 instead.

Basic DDE Functions

1. Start Excel: provides a default empty workbook, with three worksheets
2. Open an existing workbook
3. Save As of the current workbook with a new name
4. Exit Excel

All of the examples presented in the "Tour of DDE Examples" assume that Steps 1 and 2 of Basic DDE Functions have already been completed.

Starting Excel from SAS

For other versions of Microsoft Office or Excel than what I used for my development, you may need to use a folder sequence in your code different from that used in this Excel start-up code:

```
x '"C:\Program Files\Microsoft Office\Office\EXCEL.exe"';
data _null_;
z=sleep(3);  /* wait 3 seconds for Excel to start */
run;
```

The above program opens a new workbook, with three empty worksheets: Sheet1, Sheet2, and Sheet3.

Open Existing Workbook from SAS

```
data _null_;     /* talk to DDE, no output data   */
file DDEcmds;
put '[open("c:\YourFolder\YourWorkbook.xls")]';
x=sleep(3);      /* wait 3 seconds for it to open */
run;
```

Finished Workbook Save As

```
data _null_;
file DDEcmds;
put '[error(false)]';
put '[save.as ("c:\OtherFolder\DifferentFileName.xls")]';
x=sleep(1);
run;
```

Exiting Excel from SAS

```
data _null_;
file DDEcmds;
put '[error(false)]';
put '[quit()]';       /* empty parenthesis */
run;
```

The **`error(false)`** command tells Excel that you do not want a prompt to ask for confirmation of intention.

SAS-with-Excel Libraries of Programs and Macros (available via email)

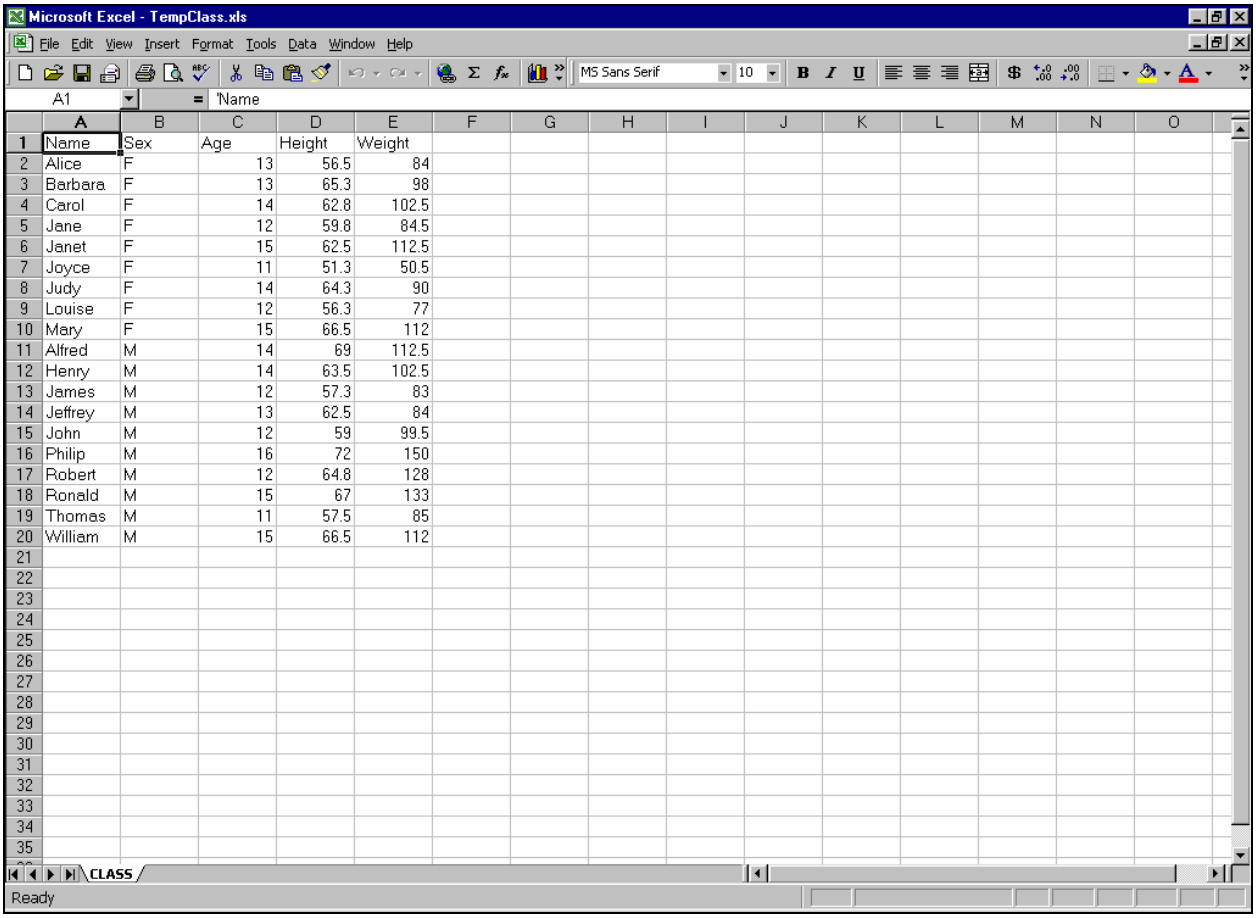
- 26 heavily commented sample programs
- 5 empty, but specially preformatted, spreadsheets to use in sample programs
- Text files of color definition information
- 60 macros for your use, with example invocations in sample programs

Running sample programs with "**`OPTIONS MPRINT;`**" shows generated SAS code in the SAS log. This enables you to understand what any macro does. If you wish to avoid use of a macro, you can copy generated code from the SAS log, strip off the MPRINT prefix at the left margin, and adapt that code to other uses without the macro.

Tour of DDE Examples

All of the examples presented in this "Tour of DDE Examples" assume that Steps 1 and 2 described in the Section "Basic DDE Functions" have already been completed.

We will use an Excel workbook which is loaded with data from SASHELP.CLASS and initially uses the default Excel formatting.



	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

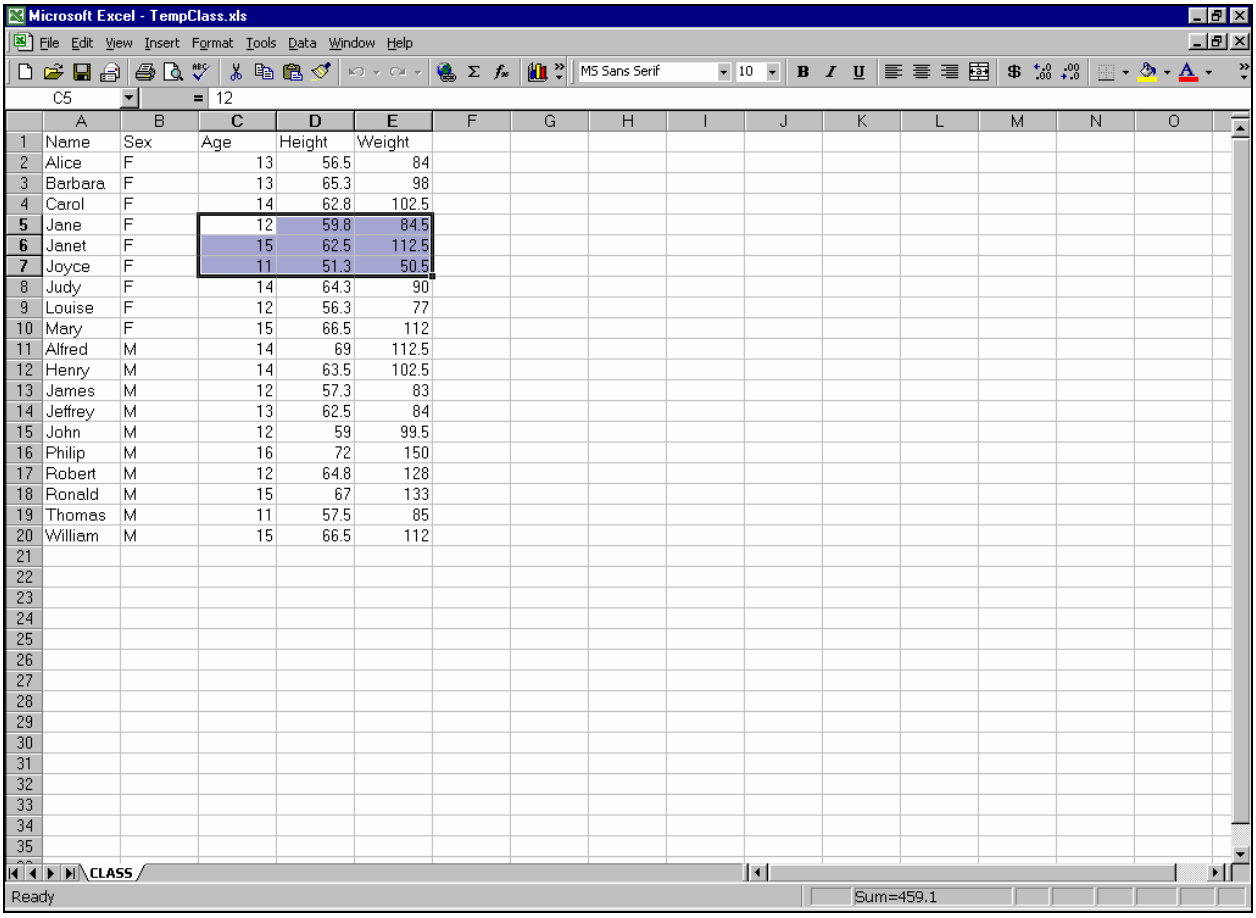
All illustrations in this paper present screen images at full page width to maximize readability. There will be pages with lots of white space at the bottom. By limiting each page to one screen image at most, it is easier to present code and its result together on the same page.

Font Formatting – Step 1

Activate worksheet (if not already there). Select the cells to be formatted

```
data _null_;
file DDEcmds;
put '[workbook.activate("CLASS")]';
put [select("R5C3:R7C5")]';
run;
```

Cells to be formatted Selected:



	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

Font Formatting – Step 2

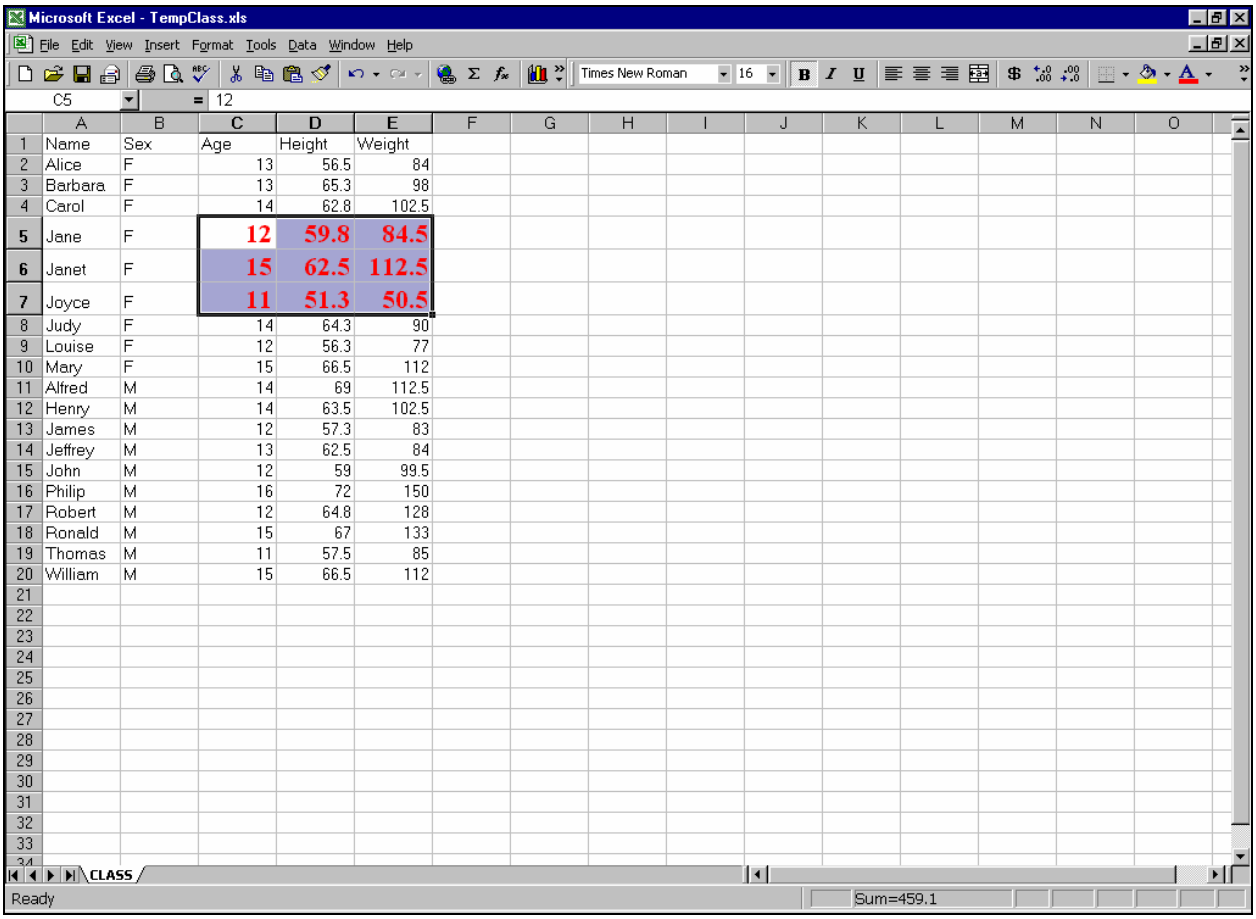
Let's use:

- font Verdana
- style Bold
- size 16 point
- (Excel) color 3 - red

Here's how (those commas are very important, and represent unused but available options):

```
data _null_;
file DDEcmds;
put '[font.properties("Verdana","Bold",16,,,,,,3)]';
run;
```

The font is changed, but is obscured by the cell selection box and shading:



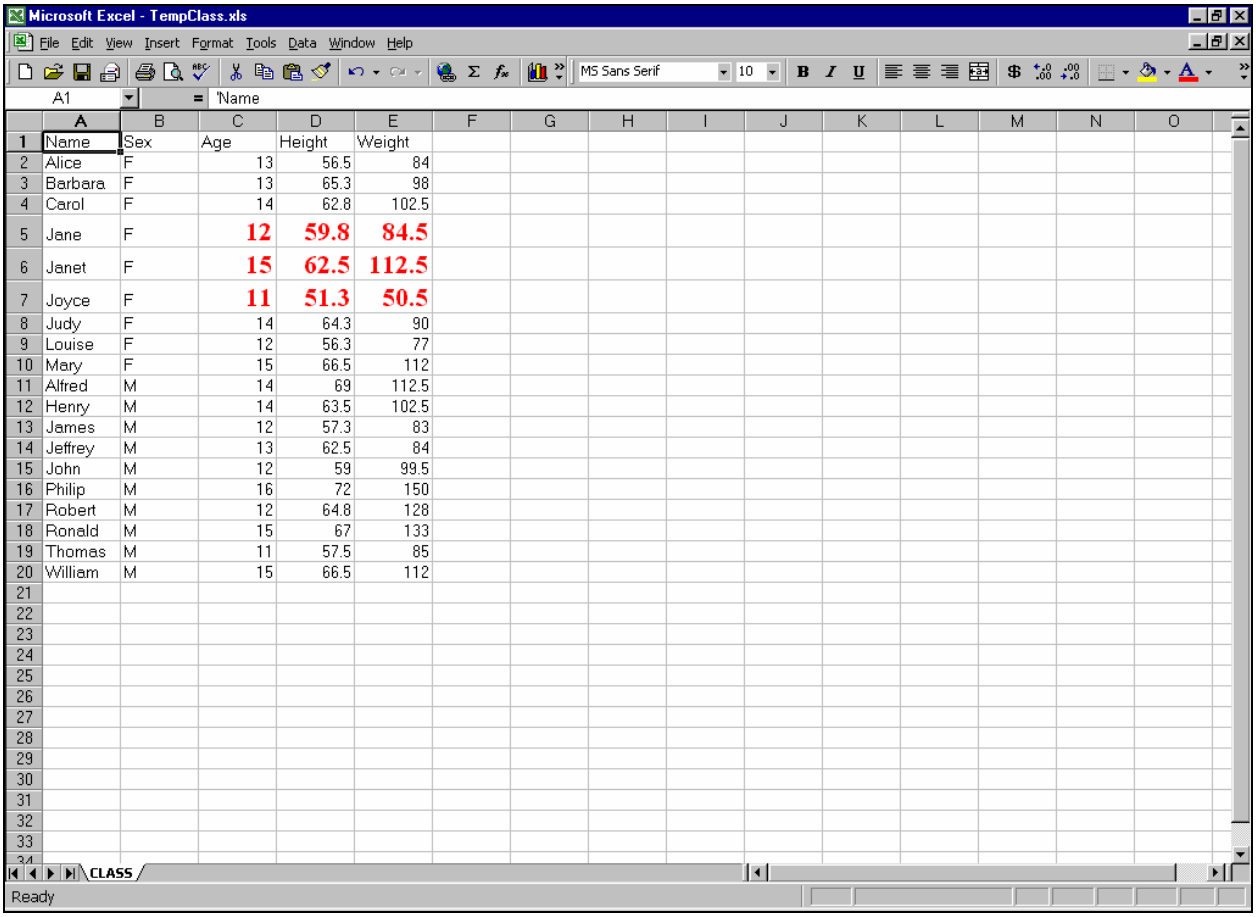
	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

Font Formatting – Step 3

After the block of cells has been formatted, the necessary predecessor **`select("R5C3:R7C5")`** command leaves them highlighted (just as a mouse would). But I want to see the new font color. So, move the focus with:

```
data _null_;
file DDEcmds;
put '[select("R1C1")]';
run;
```

"I can see clearly now."



	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

Font Formatting *Can* Be Done All In One Step

```
data _null_;
File DDEcmds;
put '[workbook.activate("Class")]';
put '[select("R5C3:R7C5")]';
put '[font.properties("Verdana","Bold Italic",16,,,,,,,3)]';
put '[select("R1C1")]';
run;
```

DDE Command Format

In a command of the form

```
put '[CommandName(arg1,arg2,...,argN)]';
```

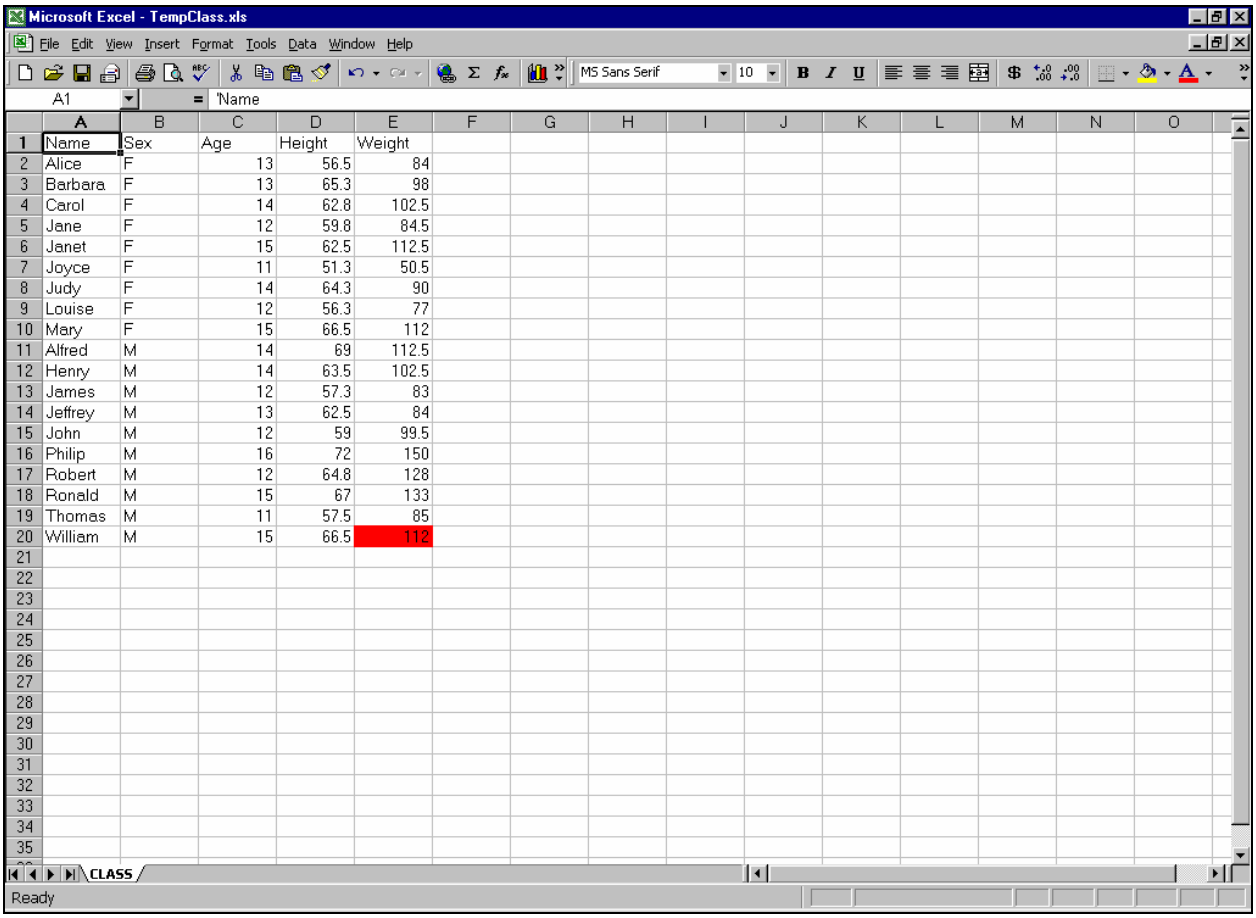
the **`CommandName`** may be multi-word, dot-separated.

For an example with several (in this case, four) command arguments, there are three ways to indicate omission of one argument—

- if at start of list: **`,arg2,arg3,arg4`**
- if in middle of list: **`arg1,,arg3,arg4`**
- if at end of list: **`arg1,arg2,arg3`** (i.e., just omit)

Color the Cell Background

```
data _null_;
file DDEcmds;
put '[select.special(11)]';  /* select the last cell of worksheet */
put '[patterns(1,,3,true)]'; /* color it red                      */
put '[select("R1C1")]';      /* move the highlight to the first cell
                                to see the color with no halo     */
run;
```

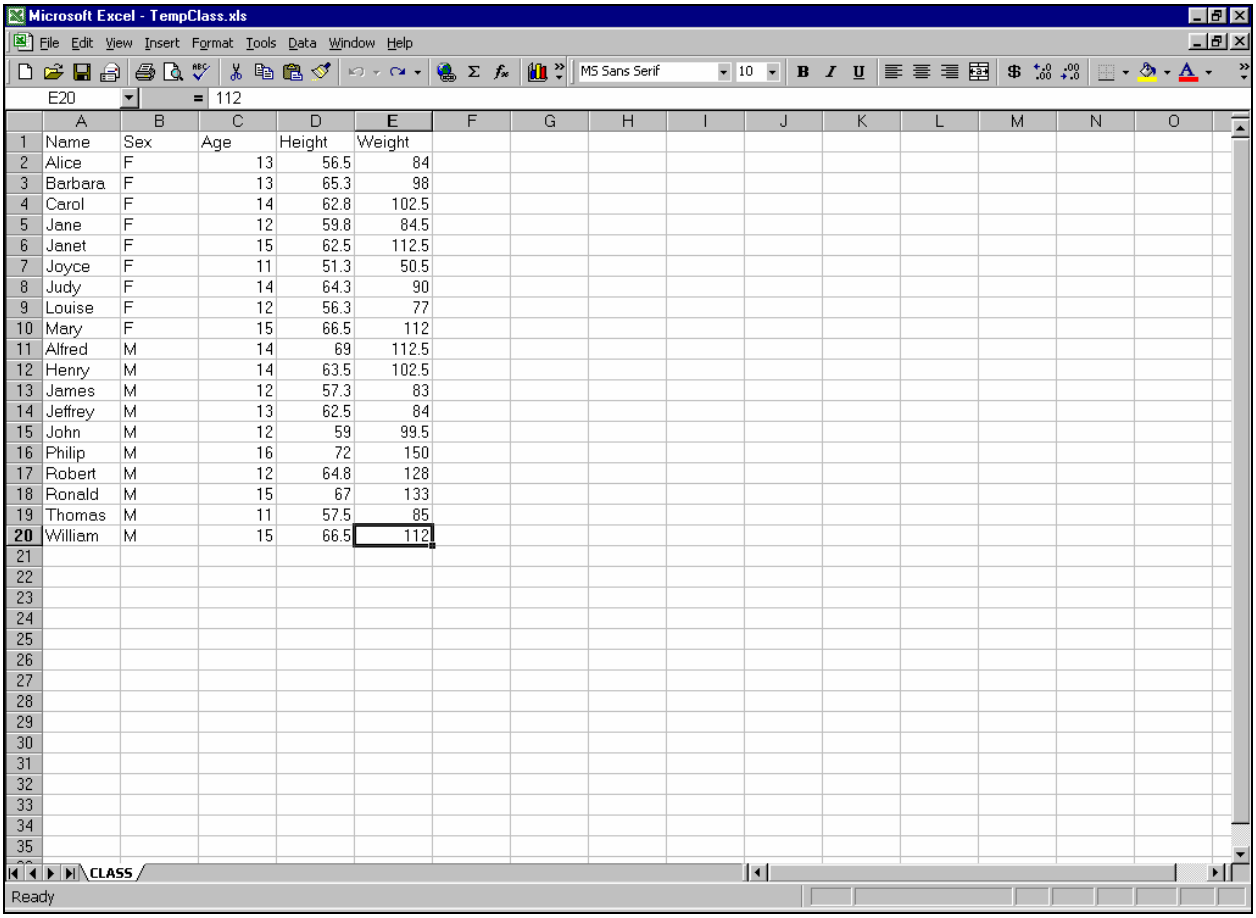


	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

Remove the Background Color
(or reverse whatever was the last command)

```
data _null_;
file DDEcmds;
put '[undo()]';
run;
```

Color Undone (but focus returns to the cell for which the prior action is undone):



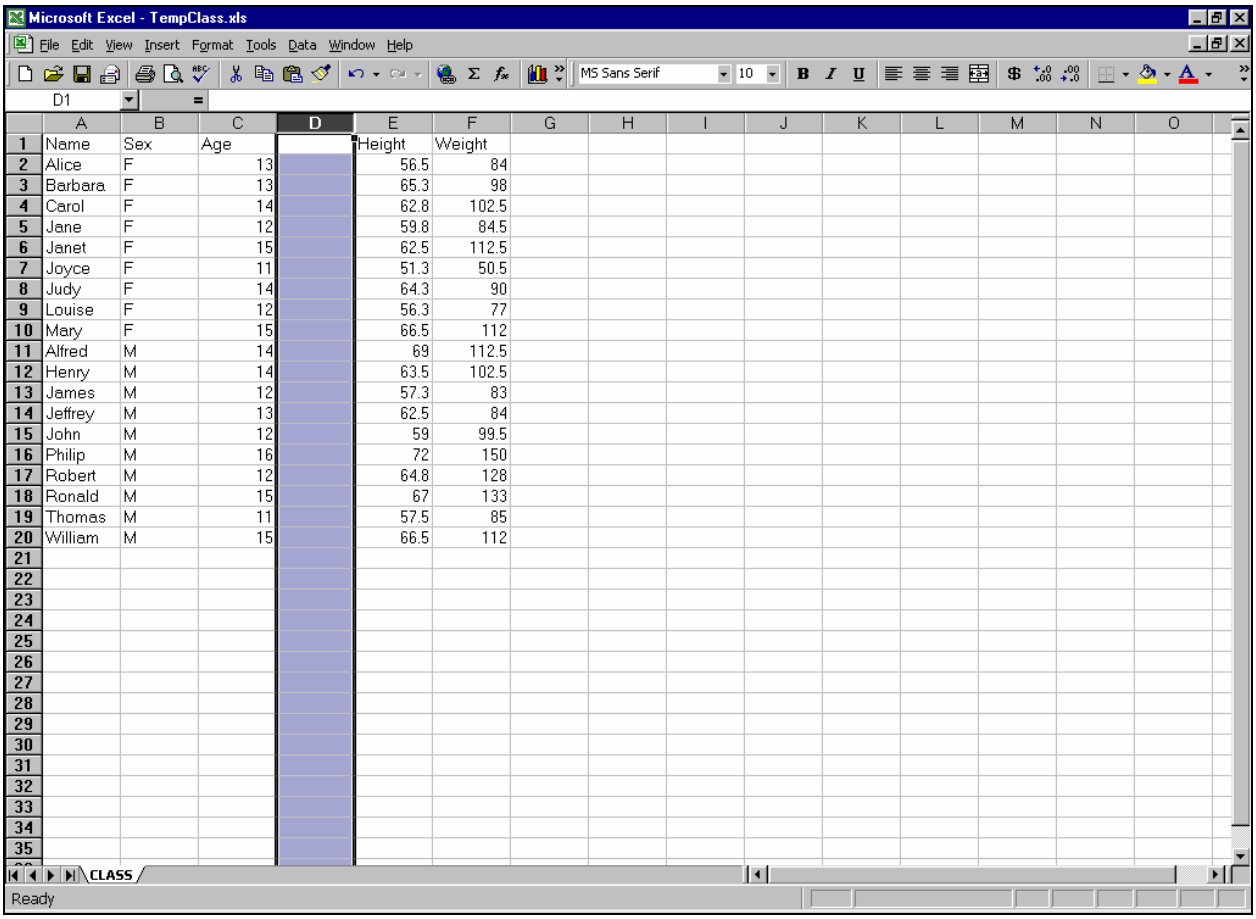
	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Jane	F	12	59.8	84.5
6	Janet	F	15	62.5	112.5
7	Joyce	F	11	51.3	50.5
8	Judy	F	14	64.3	90
9	Louise	F	12	56.3	77
10	Mary	F	15	66.5	112
11	Alfred	M	14	69	112.5
12	Henry	M	14	63.5	102.5
13	James	M	12	57.3	83
14	Jeffrey	M	13	62.5	84
15	John	M	12	59	99.5
16	Philip	M	16	72	150
17	Robert	M	12	64.8	128
18	Ronald	M	15	67	133
19	Thomas	M	11	57.5	85
20	William	M	15	66.5	112

Using Formulas in Excel

Normally, one could create all the Excel report data in SAS pre-spreadsheet-load processing. Suppose, however, we want columns derived from Excel subtotals. For demo only, we will work with simple spreadsheet content, without subtotals. With subtotals present, the use of Excel formulas works the same.

Insert a New Column to the Left of Column 4

```
data _null_;
file DDEcmds;
put '[select("C4")]';
put '[insert(4)]';    /* 4 specifies type of INSERT */
run;
```



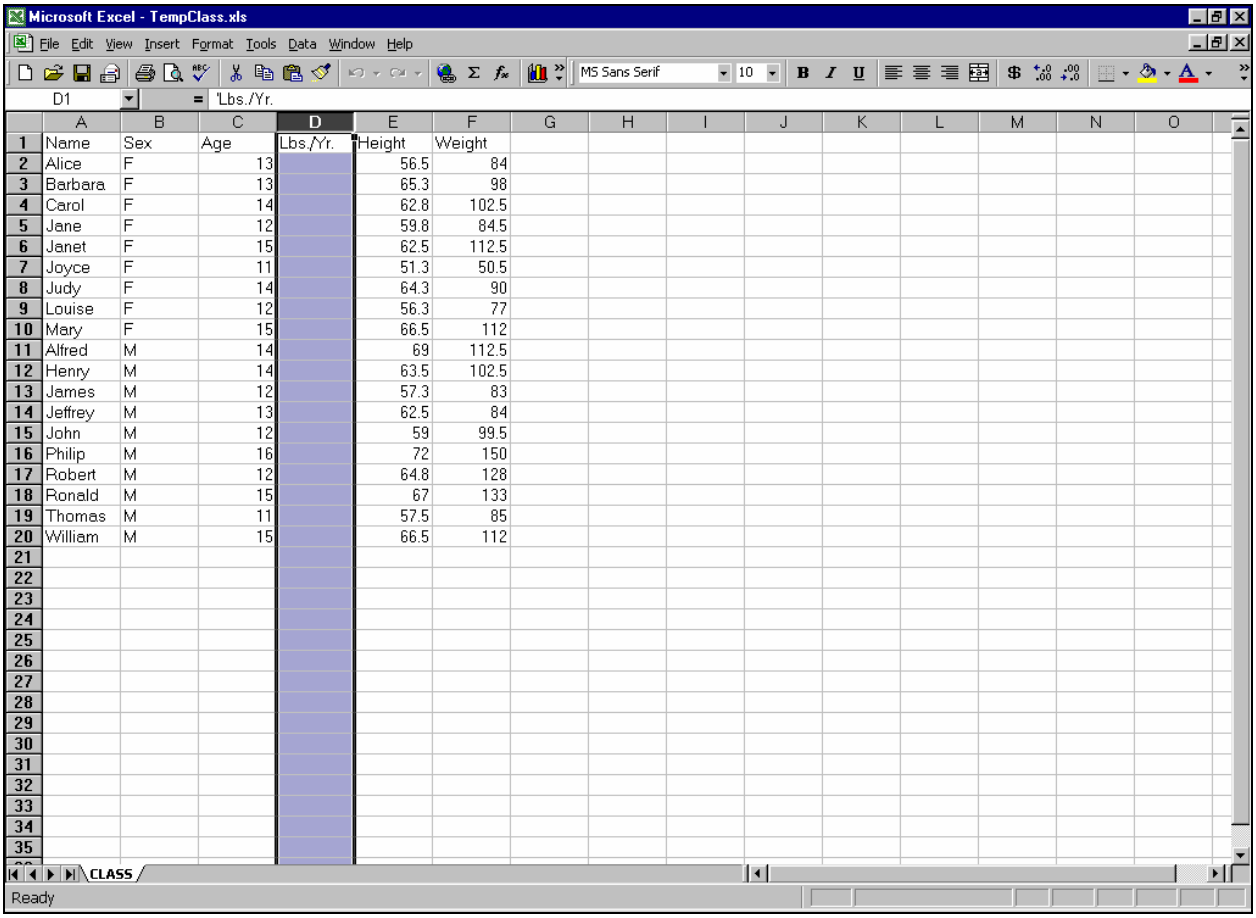
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Name	Sex	Age		Height	Weight									
2	Alice	F	13		56.5	84									
3	Barbara	F	13		65.3	98									
4	Carol	F	14		62.8	102.5									
5	Jane	F	12		59.8	84.5									
6	Janet	F	15		62.5	112.5									
7	Joyce	F	11		51.3	50.5									
8	Judy	F	14		64.3	90									
9	Louise	F	12		56.3	77									
10	Mary	F	15		66.5	112									
11	Alfred	M	14		69	112.5									
12	Henry	M	14		63.5	102.5									
13	James	M	12		57.3	83									
14	Jeffrey	M	13		62.5	84									
15	John	M	12		59	99.5									
16	Philip	M	16		72	150									
17	Robert	M	12		64.8	128									
18	Ronald	M	15		67	133									
19	Thomas	M	11		57.5	85									
20	William	M	15		66.5	112									

Excel Format the (Still Empty) Cells in Column 4 Like SAS Format 4.1

```
 /* no need to re-select the column, because focus is still there */
data _null_;
file DDEcmds;
put '[Format.Number("#0.0")]';
run;
```

Insert a Heading for the New Column 4

```
filename CELL1 DDE "EXCEL|Class!R1C4";
data _null_;
file CELL1;
put 'Lbs./Yr.';
run;
```



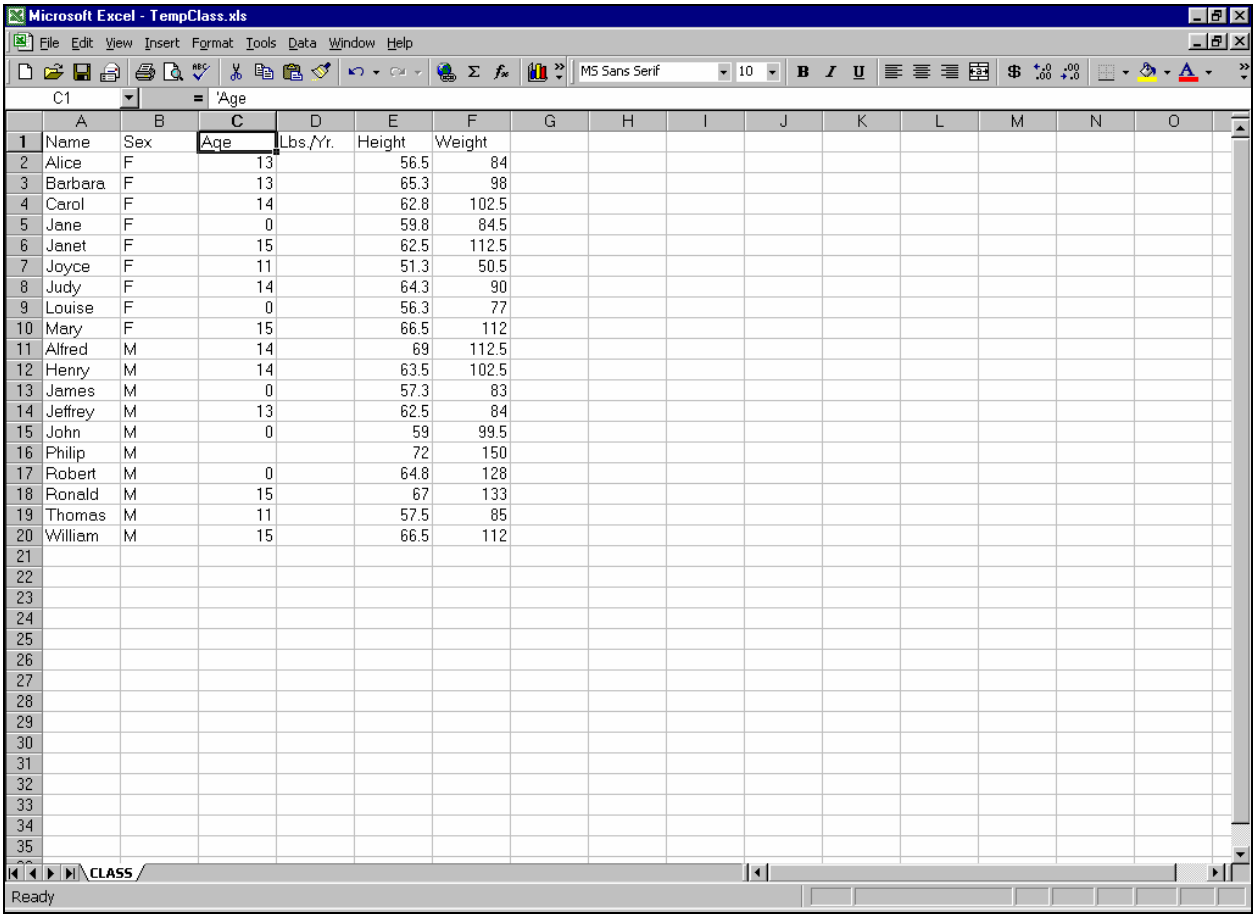
	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13		56.5	84
3	Barbara	F	13		65.3	98
4	Carol	F	14		62.8	102.5
5	Jane	F	12		59.8	84.5
6	Janet	F	15		62.5	112.5
7	Joyce	F	11		51.3	50.5
8	Judy	F	14		64.3	90
9	Louise	F	12		56.3	77
10	Mary	F	15		66.5	112
11	Alfred	M	14		69	112.5
12	Henry	M	14		63.5	102.5
13	James	M	12		57.3	83
14	Jeffrey	M	13		62.5	84
15	John	M	12		59	99.5
16	Philip	M	16		72	150
17	Robert	M	12		64.8	128
18	Ronald	M	15		67	133
19	Thomas	M	11		57.5	85
20	William	M	15		66.5	112

Prepare for a Division Operation

Column 3 is to be the divisor. For this demo, set some of its cells to null (missing) values, and some other cells to zero values.

```
data _null_;
file DDEcmds;
put '[select("C3")]';
put '[formula.replace("16","",1,,false,false)]';
put '[formula.replace("12","0",1,,false,false)]';
put '[select("R1C3")]';   /* remove highlight */
run;
```

Note the Presence Now of Some Zero Values and a Null in the Age Column:

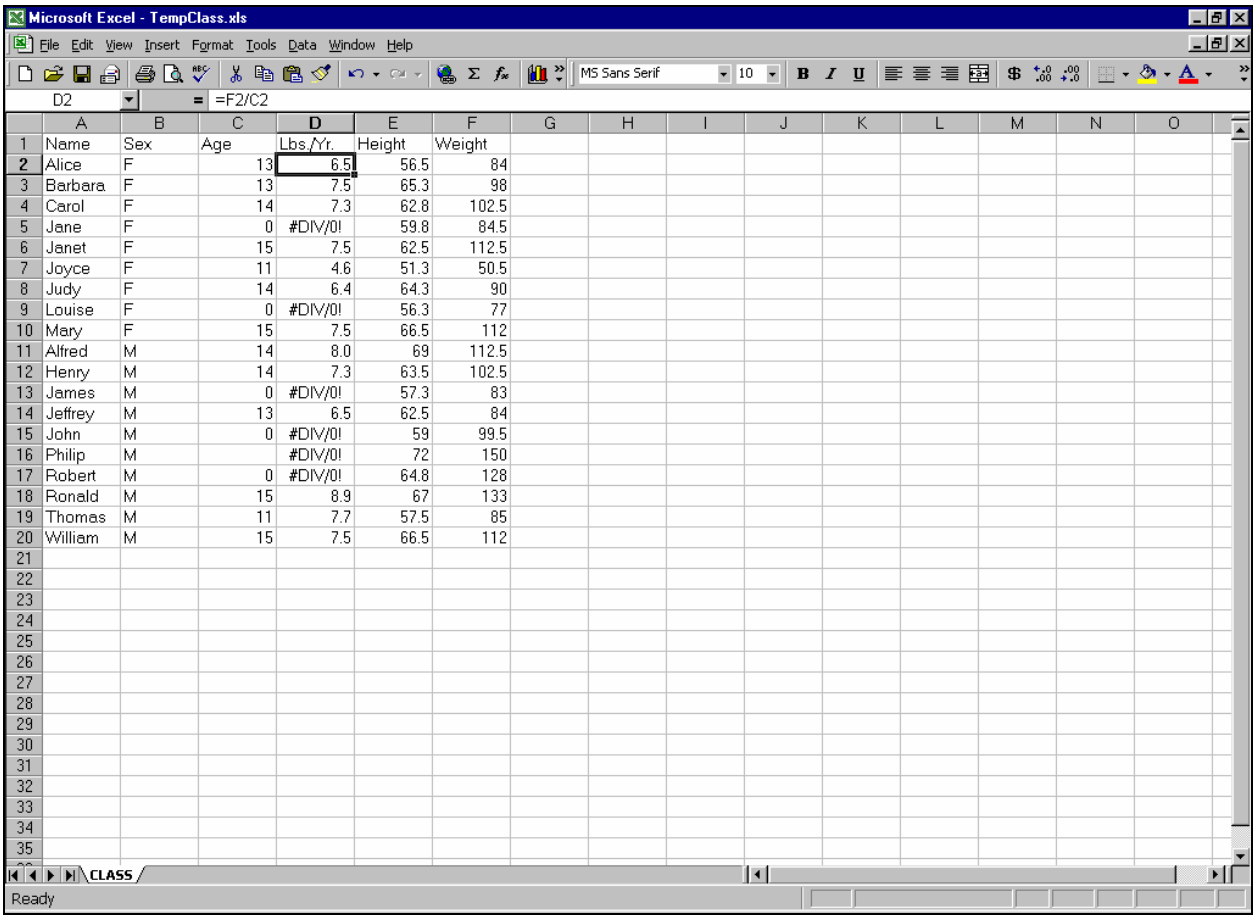


	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13		56.5	84
3	Barbara	F	13		65.3	98
4	Carol	F	14		62.8	102.5
5	Jane	F	0		59.8	84.5
6	Janet	F	15		62.5	112.5
7	Joyce	F	11		51.3	50.5
8	Judy	F	14		64.3	90
9	Louise	F	0		56.3	77
10	Mary	F	15		66.5	112
11	Alfred	M	14		69	112.5
12	Henry	M	14		63.5	102.5
13	James	M	0		57.3	83
14	Jeffrey	M	13		62.5	84
15	John	M	0		59	99.5
16	Philip	M			72	150
17	Robert	M	0		64.8	128
18	Ronald	M	15		67	133
19	Thomas	M	11		57.5	85
20	William	M	15		66.5	112

Divide Weight By Age

```
data _null_;
file DDEcmds;
put '[select("R2C4")]';
put '[formula.fill("=RC[+2]/RC[-1]","R2C4:R20C4")]';
run;
```

Some (Expected) Error Messages Appear:

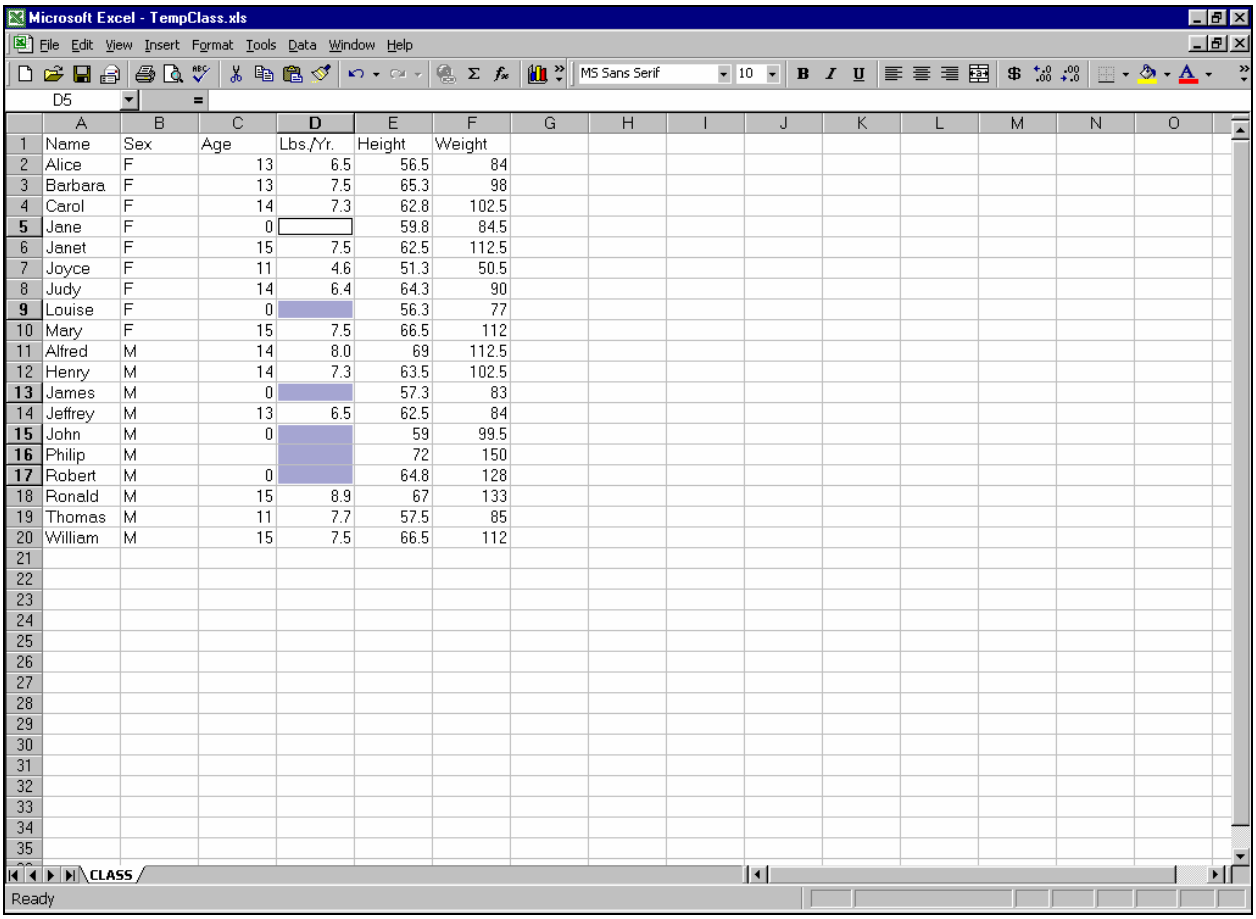


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Name	Sex	Age	Lbs./Yr.	Height	Weight									
2	Alice	F	13	6.5	56.5	84									
3	Barbara	F	13	7.5	65.3	98									
4	Carol	F	14	7.3	62.8	102.5									
5	Jane	F	0	#DIV/0!	59.8	84.5									
6	Janet	F	15	7.5	62.5	112.5									
7	Joyce	F	11	4.6	51.3	50.5									
8	Judy	F	14	6.4	64.3	90									
9	Louise	F	0	#DIV/0!	56.3	77									
10	Mary	F	15	7.5	66.5	112									
11	Alfred	M	14	8.0	69	112.5									
12	Henry	M	14	7.3	63.5	102.5									
13	James	M	0	#DIV/0!	57.3	83									
14	Jeffrey	M	13	6.5	62.5	84									
15	John	M	0	#DIV/0!	59	99.5									
16	Philip	M		#DIV/0!	72	150									
17	Robert	M	0	#DIV/0!	64.8	128									
18	Ronald	M	15	8.9	67	133									
19	Thomas	M	11	7.7	57.5	85									
20	William	M	15	7.5	66.5	112									
21															
22															
23															
24															
25															
26															
27															
28															
29															
30															
31															
32															
33															
34															
35															

Get Rid of the Error Message #DIV/0!

```
data _null_;
file DDEcmds;
put '[select.special(3,16)]';  /* select cells with error messages */
put '[clear]';
run;
```

Cleared Cells Remain Selected:

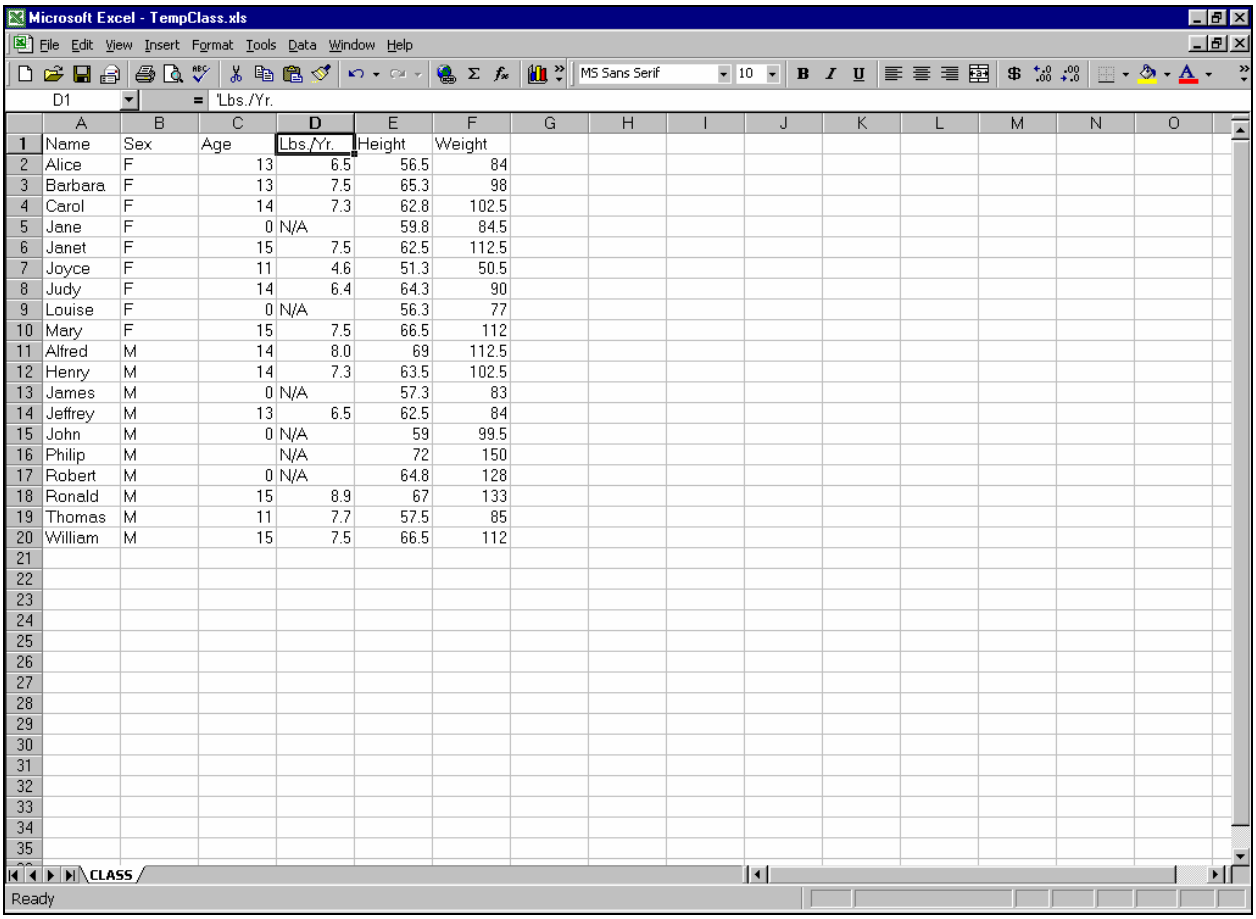


	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13	6.5	56.5	84
3	Barbara	F	13	7.5	65.3	98
4	Carol	F	14	7.3	62.8	102.5
5	Jane	F	0		59.8	84.5
6	Janet	F	15	7.5	62.5	112.5
7	Joyce	F	11	4.6	51.3	50.5
8	Judy	F	14	6.4	64.3	90
9	Louise	F	0		56.3	77
10	Mary	F	15	7.5	66.5	112
11	Alfred	M	14	8.0	69	112.5
12	Henry	M	14	7.3	63.5	102.5
13	James	M	0		57.3	83
14	Jeffrey	M	13	6.5	62.5	84
15	John	M	0		59	99.5
16	Philip	M			72	150
17	Robert	M	0		64.8	128
18	Ronald	M	15	8.9	67	133
19	Thomas	M	11	7.7	57.5	85
20	William	M	15	7.5	66.5	112

Replace Nulls with "N/A"

```
data _null_;
file DDEcmds;
put '[formula.replace("","N/A",1,,false,false)]';
put '[select("R1C4")]';  /* remove the highlight */
run;
```

Almost Finished:

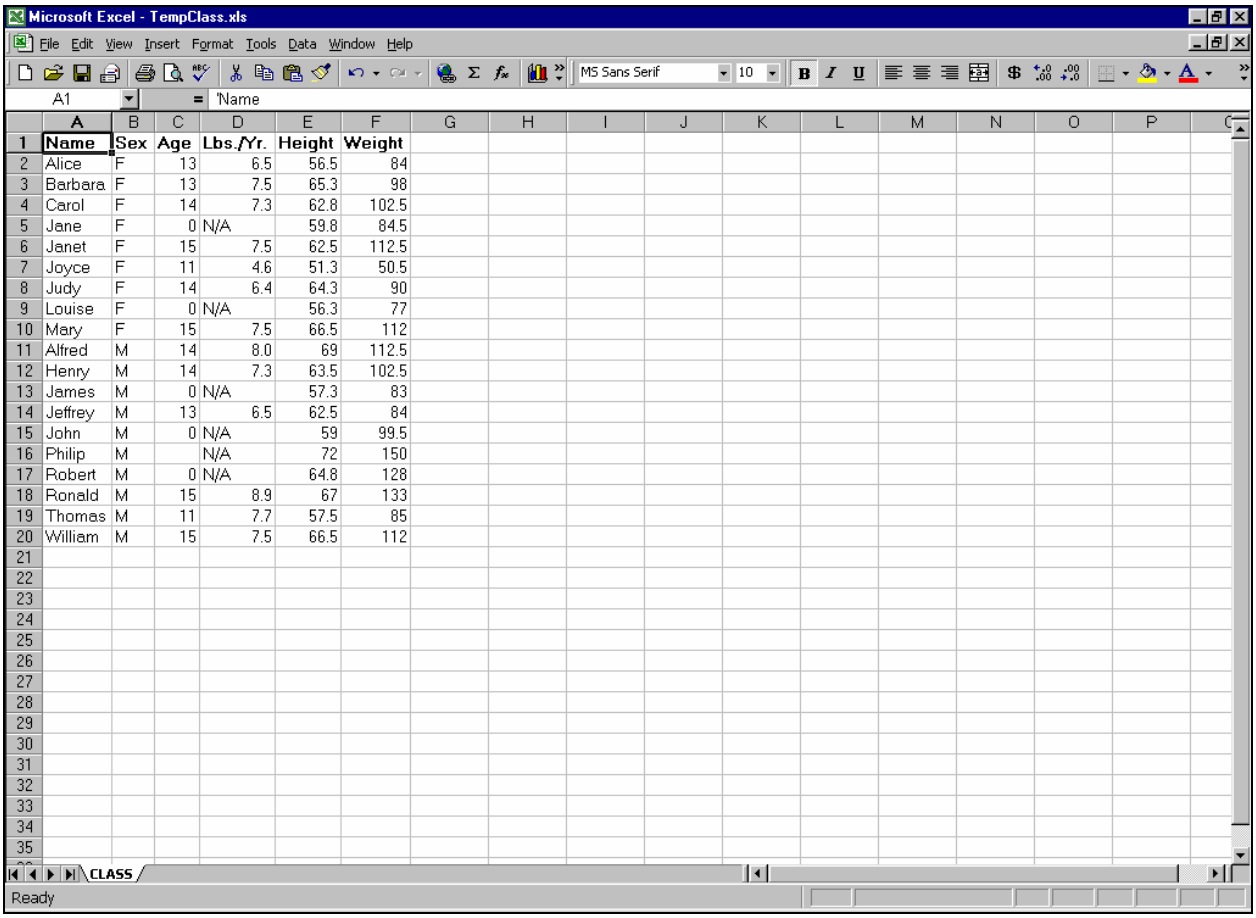


	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13	6.5	56.5	84
3	Barbara	F	13	7.5	65.3	98
4	Carol	F	14	7.3	62.8	102.5
5	Jane	F	0	N/A	59.8	84.5
6	Janet	F	15	7.5	62.5	112.5
7	Joyce	F	11	4.6	51.3	50.5
8	Judy	F	14	6.4	64.3	90
9	Louise	F	0	N/A	56.3	77
10	Mary	F	15	7.5	66.5	112
11	Alfred	M	14	8.0	69	112.5
12	Henry	M	14	7.3	63.5	102.5
13	James	M	0	N/A	57.3	83
14	Jeffrey	M	13	6.5	62.5	84
15	John	M	0	N/A	59	99.5
16	Philip	M		N/A	72	150
17	Robert	M	0	N/A	64.8	128
18	Ronald	M	15	8.9	67	133
19	Thomas	M	11	7.7	57.5	85
20	William	M	15	7.5	66.5	112

Make the Headings Bold, and AutoFit the Columns

```
data _null_;
file DDEcmds;
put '[select("R1")]';
put '[font.properties("","Bold")]';
                    /* "", = font name unchanged          */
put '[select("C1:C6")]';
put '[column.width(,,,3)]'; /* commas required            */
                    /* 3 = AutoFit                        */
put '[select("R1C1")]'; /* remove the selection highlight */
run;
```

Better, But Still Not Finished:

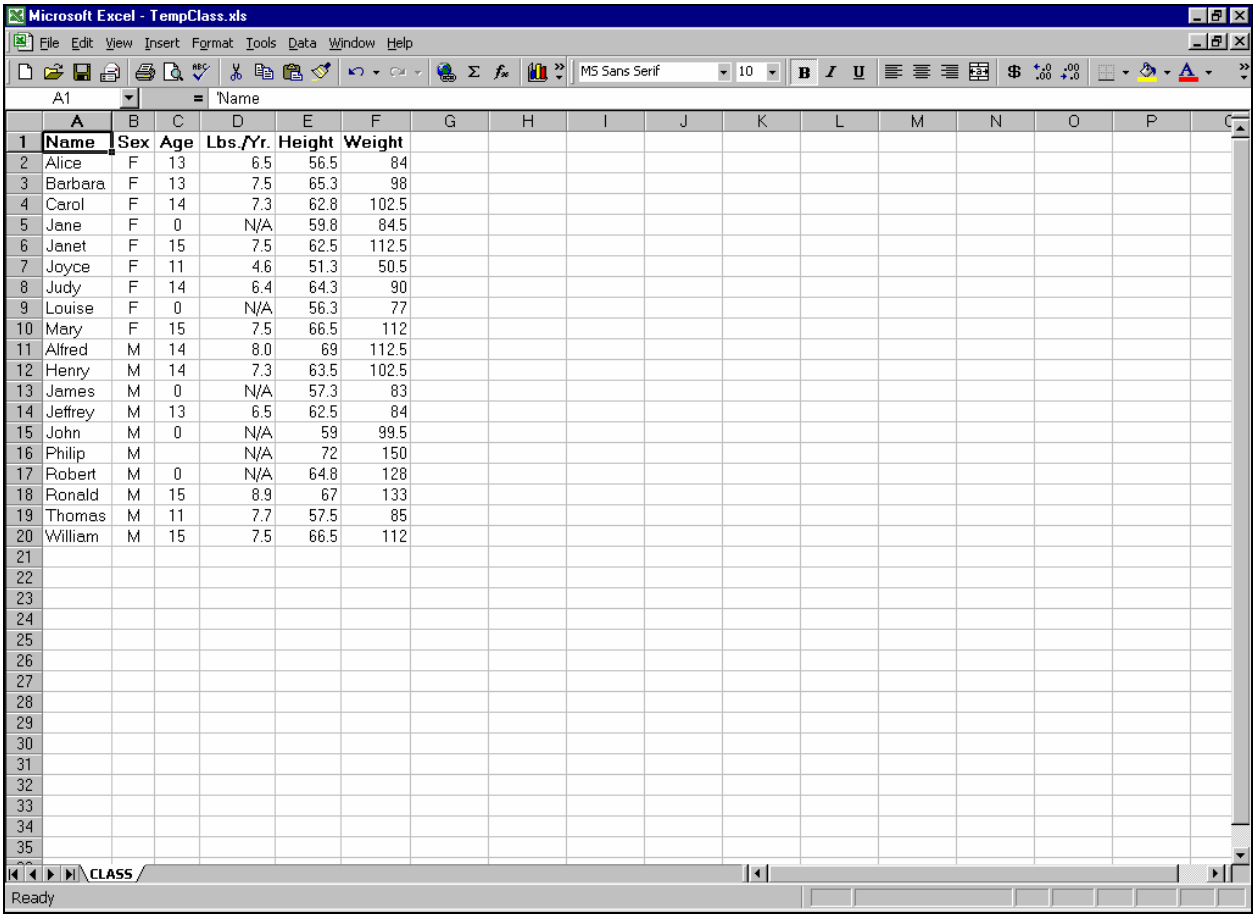


	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13	6.5	56.5	84
3	Barbara	F	13	7.5	65.3	98
4	Carol	F	14	7.3	62.8	102.5
5	Jane	F	0	N/A	59.8	84.5
6	Janet	F	15	7.5	62.5	112.5
7	Joyce	F	11	4.6	51.3	50.5
8	Judy	F	14	6.4	64.3	90
9	Louise	F	0	N/A	56.3	77
10	Mary	F	15	7.5	66.5	112
11	Alfred	M	14	8.0	69	112.5
12	Henry	M	14	7.3	63.5	102.5
13	James	M	0	N/A	57.3	83
14	Jeffrey	M	13	6.5	62.5	84
15	John	M	0	N/A	59	99.5
16	Philip	M		N/A	72	150
17	Robert	M	0	N/A	64.8	128
18	Ronald	M	15	8.9	67	133
19	Thomas	M	11	7.7	57.5	85
20	William	M	15	7.5	66.5	112

Align Columns 2 and 3

```
data _null_;
file DDEcmds;
put '[select("C2:C3")]';          /* Sex & Age Columns */
put '[alignment(3,false,3,0)]';   /* 3 = Center        */
put '[select("C4")]';             /* Lbs./Yr. Column   */
put '[alignment(4, false,3,0)]'; /* 4 = Right         */
put '[select("R1C1")]';
run;
```

But Need To Fix the Format of Height and Weight:

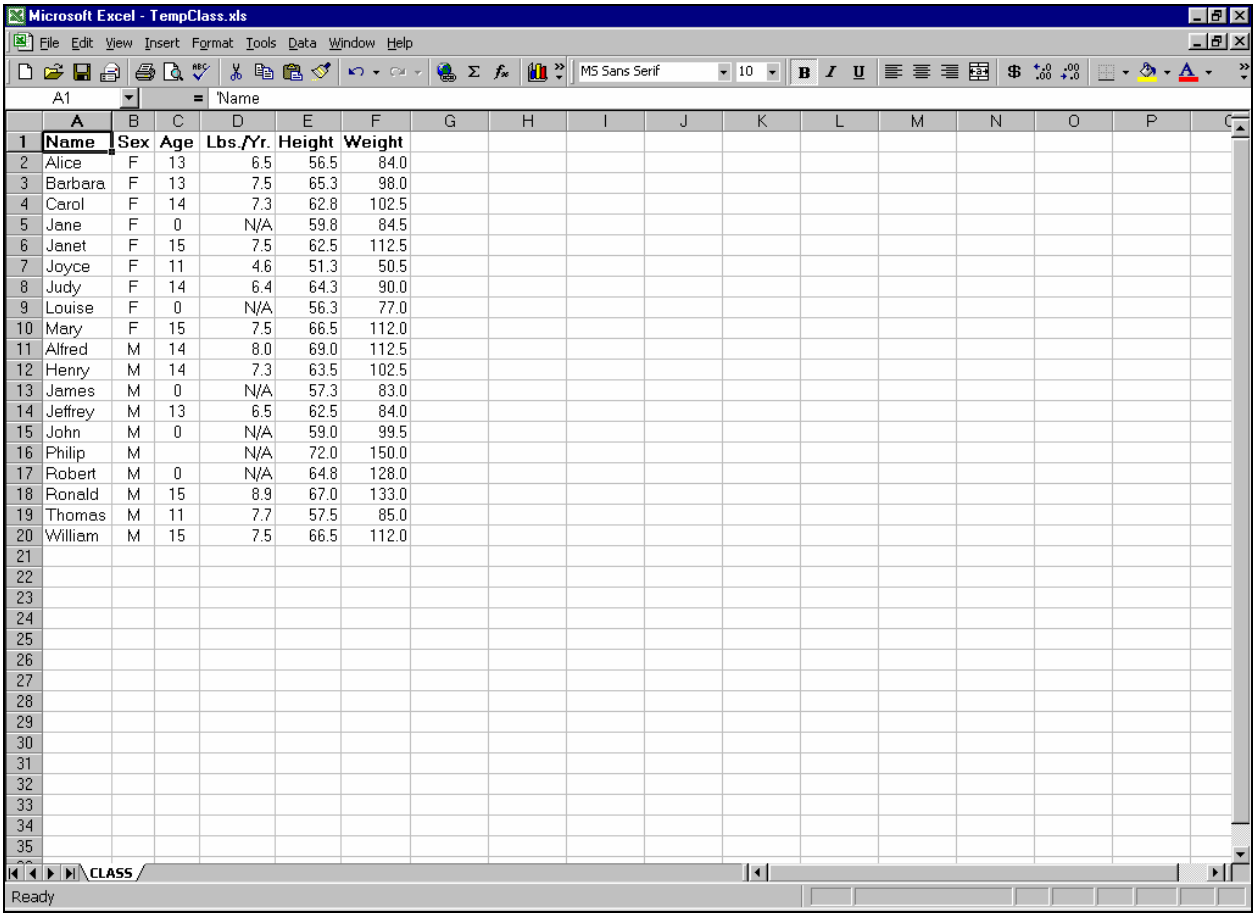


	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13	6.5	56.5	84
3	Barbara	F	13	7.5	65.3	98
4	Carol	F	14	7.3	62.8	102.5
5	Jane	F	0	N/A	59.8	84.5
6	Janet	F	15	7.5	62.5	112.5
7	Joyce	F	11	4.6	51.3	50.5
8	Judy	F	14	6.4	64.3	90
9	Louise	F	0	N/A	56.3	77
10	Mary	F	15	7.5	66.5	112
11	Alfred	M	14	8.0	69	112.5
12	Henry	M	14	7.3	63.5	102.5
13	James	M	0	N/A	57.3	83
14	Jeffrey	M	13	6.5	62.5	84
15	John	M	0	N/A	59	99.5
16	Philip	M		N/A	72	150
17	Robert	M	0	N/A	64.8	128
18	Ronald	M	15	8.9	67	133
19	Thomas	M	11	7.7	57.5	85
20	William	M	15	7.5	66.5	112

Excel Format the Height and Weight Data Like SAS Format 4.1

```
data _null_;
file DDEcmds;
put '[select("C5:C6")]';
put '[format.number("#0.0")]';
put '[select("R1C1")]';        /* remove the column selection
highlight */
run;
```

Finished:



	A	B	C	D	E	F
1	Name	Sex	Age	Lbs./Yr.	Height	Weight
2	Alice	F	13	6.5	56.5	84.0
3	Barbara	F	13	7.5	65.3	98.0
4	Carol	F	14	7.3	62.8	102.5
5	Jane	F	0	N/A	59.8	84.5
6	Janet	F	15	7.5	62.5	112.5
7	Joyce	F	11	4.6	51.3	50.5
8	Judy	F	14	6.4	64.3	90.0
9	Louise	F	0	N/A	56.3	77.0
10	Mary	F	15	7.5	66.5	112.0
11	Alfred	M	14	8.0	69.0	112.5
12	Henry	M	14	7.3	63.5	102.5
13	James	M	0	N/A	57.3	83.0
14	Jeffrey	M	13	6.5	62.5	84.0
15	John	M	0	N/A	59.0	99.5
16	Philip	M		N/A	72.0	150.0
17	Robert	M	0	N/A	64.8	128.0
18	Ronald	M	15	8.9	67.0	133.0
19	Thomas	M	11	7.7	57.5	85.0
20	William	M	15	7.5	66.5	112.0

Conclusion

There are other ways to generate Excel reports with SAS, but DDE is the method I prefer. Its astonishing power has impressed me. The examples above are only a small subset of what you can do with DDE. Though still just a larger subset of DDE's power, available at no charge via email at the address below are my 60 macros, 26 sample programs (commented), and supporting files, to implement the above and other functions, including:

- Cell sizing
- Column or row Hide/Unhide
- Text wrap in cells

- Create borders to section the worksheet
- Indent cell content
- Freeze/Unfreeze panes
- Turn AutoFilter On/Off
- Generate subtotals in a worksheet
- Control/Change levels for Excel subtotals (or create custom worksheets for various levels of totals)
- Create additional worksheets in a workbook, Delete unused worksheets in a workbook
- Delete columns or rows
- Copy cells, Move cells
- Use Zoom to shrink or enlarge a worksheet
- Control the Message text box in the lower left corner of the Excel window
- Run Excel macros

Bibliography (Including a Paper on Use of DDE for Microsoft Word)

1. Koen Vyverman, "Using Dynamic Data Exchange to Export Your SAS Data to MS Excel – Against All ODS, Part I", *Proceedings of the Twenty-Sixth Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC, USA), 2001.

2. Koen Vyverman, "Creating Custom Excel Workbooks from Base SAS with Dynamic Data Exchange", *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC, USA), 2002.

3. Phil Mason, "Customising ODS Output for Microsoft Word Using DDE", *VIEWS 2003 Proceedings*, London, UK, 2003. Available at http://www.views-uk.demon.co.uk/Conference/paper_requests.htm.

4. LeRoy Bessler, "SAS and Excel, A Winning Combination, Part 1: Easy Solutions for Multi-Sheet Excel Workbooks, Color-Coding, Imbedded Graphs, etc.", *Proceedings of the MidWest SAS Users Group Conference 2010*, MWSUG, Inc. (USA), 2010.

Acknowledgments

My thanks to Craig Wildeman for explaining how to create a PDF from Excel 2007 with SAS using DDE and to Alexandra Riley for confirming that DDE also works with Excel 2010.

Author Information

Your requests, questions, comments, suggestions, and other DDE solutions are welcome.

LeRoy Bessler PhD
Bessler Consulting and Research, Mequon, Milwaukee, Wisconsin, USA
Le_Roy_Bessler@wi.rr.com

Dr. LeRoy Bessler has been a speaker at software user conferences in the US, Canada, and Europe. His research and development work is focused on communication-effective visual communication and reporting. Other interests are web information delivery, highly formatted Excel reporting with SAS, tools to support servers and users, and application development methods that maximize Reliability, Reusability, Extendibility, and Maintainability. His favorite tools are SAS/GRAPH, ODS, Base SAS, and SAS Macro Language. A SAS user since 1978, LeRoy has supported SAS users, software, servers, and data, and has done SAS application development, both as an employee and as a consultant. He is a regular contributor to *VIEWS News*, the web newsletter of the VIEWS International SAS Programmer Community.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.
® indicates USA registration. Other product and brand names are trademarks or registered trademarks of their respective owners.

Appendix 1: Important Information about Using DDE for SAS with Excel

If you use DDE for SAS-with-Excel on your desktop/laptop, you may not have any problems. You have control of all Excel options, and you can see everything that is happening with SAS and Excel.

If you have to use SAS and Excel on a remote server, there are some things to be aware of.

1. Possible SAS Situation

If you are using SAS Enterprise Guide as a client for that remote server and your code has a malfunction, a SAS process may remain running or "hung" on the server. If your version of SAS EG permits you to kill the process, you may not have a problem. If your version of SAS EG does not permit you to kill the process, and you do not have Remote Desktop or some sort of terminal access to the server, you will need to have a Windows System Administrator kill it for you. **NOTE:** I recently developed SAS macro tools to allow an Enterprise Guide user to terminate a hung SAS process. For details, please see *http://support.sas.com/resources/papers/proceedings10/279-2010.pdf*

2. Possible Excel Situation

If your code has a malfunction, an Excel session may be left open on the server. If it had an Excel file open, the file will remain open. If you do not have Remote Desktop or some sort of terminal access to the server, you will need to have a Windows System Administrator close the file and terminate the Excel session for you. **NOTE:** The SAS macro tools mentioned in Item 1 above can be easily adapted to terminate an Excel process on a remote server. If you need an explanation, please send me an email.

3. Possible Excel Situation

Only two of the examples involve the use of a custom Excel macro. Excel on your server may be set to inhibit the use of such Excel macros, or may put certain conditions on their use. If so, you need to work with your Windows System Administrator either to get the inhibition removed or to find out what you need to do get your macro(s) authorized to run.

Appendix 2: Using Dynamic Data Exchange to Load a Pivot Table from SAS with Your SAS-prepared Data

Abstract

This coding tidbit shows you how to load a pivot table with SAS by running a pre-recorded custom Excel macro via DDE.

Introduction

Knowing that pivot tables are a very common destination of SAS-originated data, whether prepared with PROC SUMMARY or some other method, and being a DDE enthusiast, I long wanted to use DDE to create pivot tables. There appears to be some machinery available, if you read MacroFun.HLP, but I could not get it to work. When I asked Cynthia Zender from SAS Institute about my problem, she suggested pre-recording an Excel macro to handle the load, and then running the Excel macro with DDE. I had a little bit of experience with that, and so this solution is the result.

For a different way to create pivot tables from SAS, please see Conway's work in Reference 1.

Solution

The code below is meant to be self-explanatory, and it assumes that you have already worked through other information, including the main body of the paper and the more basic sample programs that are in the BesslerSASandExcel zip file package available via email. It uses SAS macros from that package.

```
* UseCustomPreRecordedPivotTableMacro.sas *;

* NOTE:
  To use this sample code,
  you can choose to highlight specific code blocks
  and submit them individually.
  This sample program need not be submitted and run en masse.
  Once Excel has been started and a spreadsheet has been opened,
  you can toggle over to the Excel window
  after each SAS code execution step
  to inspect the results. *;

* These folders can, of course, be named anything you prefer,
  but the expected macros, input data, or output data
  must be stored there. *;
%let FolderForMacros    = C:\BesslerSASandExcel\macros;
%let FolderForXLSfiles  = C:\BesslerSASandExcel\spreadsheets;
%let FolderForXLSoutput = C:\BesslerSASandExcel\output;

* tell SAS where the macros are stored *;
filename SASandXL "&FolderForMacros";
* search for macros in the order listed *;
options sasautos=(SASandXL sasautos);

options mprint; * list the resolved translation of any macros
                  in the SAS log *;

* Specifying the location of the Excel executable *;
%XLStart(ExcelPgm=C:\Program Files\Microsoft Office\Office10\EXCEL.exe);

%XLDDEcmdsFileName;
```

```
* open the workbook that contains the macro(s) *;
%XLOpen(WorkBook=&FolderForXLSfiles.\PivotTableMacroForSASHELPdotCLASSweightVar.xls);

* open the workbook to be processed with the macro(s) *;
%XLOpen(WorkBook=&FolderForXLSfiles.\TempClass.xls);

* run the macro *;
* The DDE Pivot function refused to work for this situation.
  that's why I use the pre-recorded macro instead. *;
%XLRunMacro(WorkBook=PivotTableMacroForSASHELPdotCLASSweightVar.xls,
            XLMacro=Macro1);

%XLDeleteWorkSheet(WorkSheet=CLASS);

* save the result with a new XLS file name *;
%XLSaveAs(WorkBook=&FolderForXLSoutput.\sasHelpClassWeightVarInPivotTable.xls);

* Close that WorkBook *;
%XLClose;

* Close the Excel Macro WorkBook *;
%XLClose;

* Exit *;
%XLExit;
```

Reference (Non-DDE Solution)

1. Ted Conway, Sur La Table: Creating Microsoft Excel PivotTables in a Jiffy from SAS Data, *Proceedings of the Thirty-First Annual SAS Users Group International Conference*, 2006. Cary, NC: SAS Institute Inc.

Appendix 3: Using DDE to Insert and Re-Size SAS-Created Graphs

Abstract

These sample programs show you how to use DDE to insert (and, if desired, re-size) graphs, either standalone on separate worksheets, in groups on a worksheet, or with a companion table.

Introduction

I have often been asked how to use DDE to create Excel worksheets that contain SAS/GRAPH output, and always had to say that I did not know. Since in other papers I have been exploring and explaining how to create SAS graphs in a wide variety of different delivery contexts, I finally decided to investigate Excel via DDE. Here no use is made of my SAS-DDE macros. All of the programming is explicit and hard-coded. The new DDE tools used are the INSERT.PICTURE and FORMAT.SIZE functions.

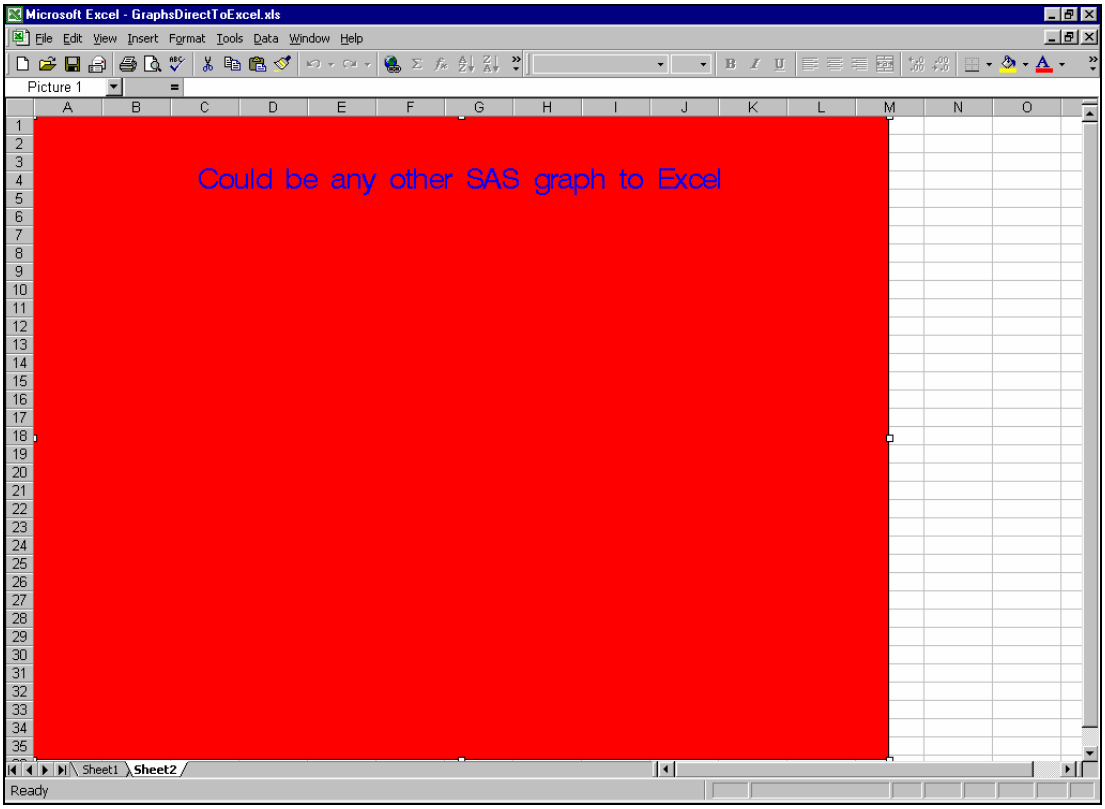
Examples of SAS/GRAPH to Excel via DDE

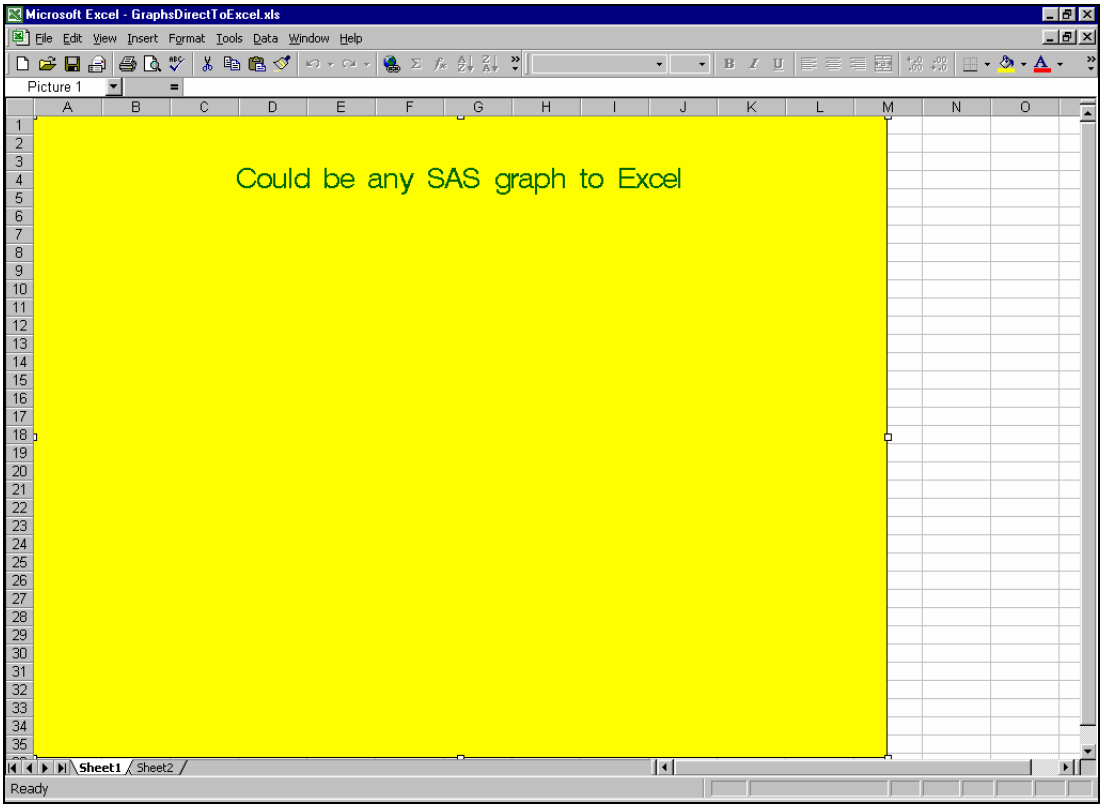
The code below is meant to be understandable without further explanation, and assumes that you already understand basic DDE functions.

In all three examples, a new Excel workbook is opened, one worksheet is customized, and the remaining two sheets present by default are deleted. Instead, a pre-existing workbook could be used and customized, by modifying a pre-existing worksheet or a newly inserted one.

Since these examples are meant to demonstrate how to get SAS/GRAPH output into Excel, and not to teach you how to use SAS/GRAPH, the “graphs” are merely text slides that are trivial to create.

Two Graphs to Two Separate Worksheets

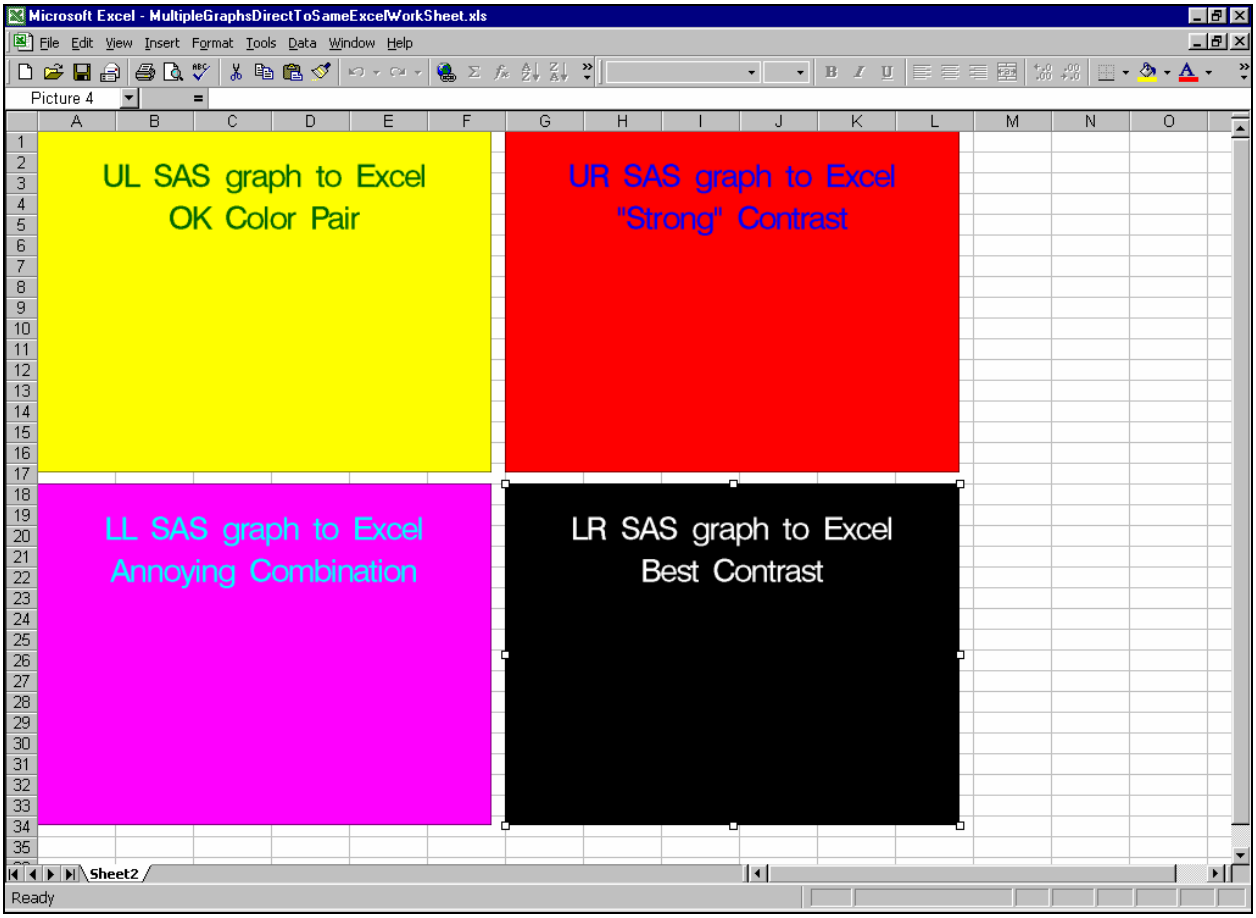




Here is the code used to create the Excel workbook above:

```
goptions reset=all;
goptions device=GIF;
goptions border; /* Put the graph in a box. */
goptions gsfmode=replace gsfname=anyname;
proc gslide;
filename anyname "C:\image1.GIF";
goptions cback=CXFFFF00; /* Use RGB Yellow for background. */
  /* Use RGB Medium Dark Green for title. */
title1 ' ' justify=CENTER height=2 color=CX006600 'Could be any SAS graph to Excel';
run;
filename anyname "C:\image2.GIF";
goptions cback=CXFF0000; /* Use RGB Red for background. */
  /* Use RGB Blue for text. */
title1 ' ' justify=CENTER height=2 color=CX0000FF 'Could be any other SAS graph to
Excel';
run;
quit;

options noxwait noxsync;

x '"C:\Program Files\Microsoft Office\Office\EXCEL.exe"';

data _null_;
z=sleep(3); /* wait 3 seconds for Excel to start */
run;

filename DDEcmds dde "excel|system";

data _null_;
file DDEcmds;
put '[workbook.activate("Sheet1")]';
put '[insert.picture("C:\image1.GIF")]';
put '[format.size(600,450)]'; /* this really keeps it unchanged */
put '[workbook.activate("Sheet2")]';
put '[insert.picture("C:\image2.GIF")]';
put '[error(false)]';
put '[workbook.delete("Sheet3")]';
put '[error(false)]';
put '[save.as("C:\GraphsDirectToExcel.xls")]';
z=sleep(3); /* wait 3 seconds for Save to end */
put '[error(false)]';
put '[quit()]'; /* empty parenthesis */
run;
```

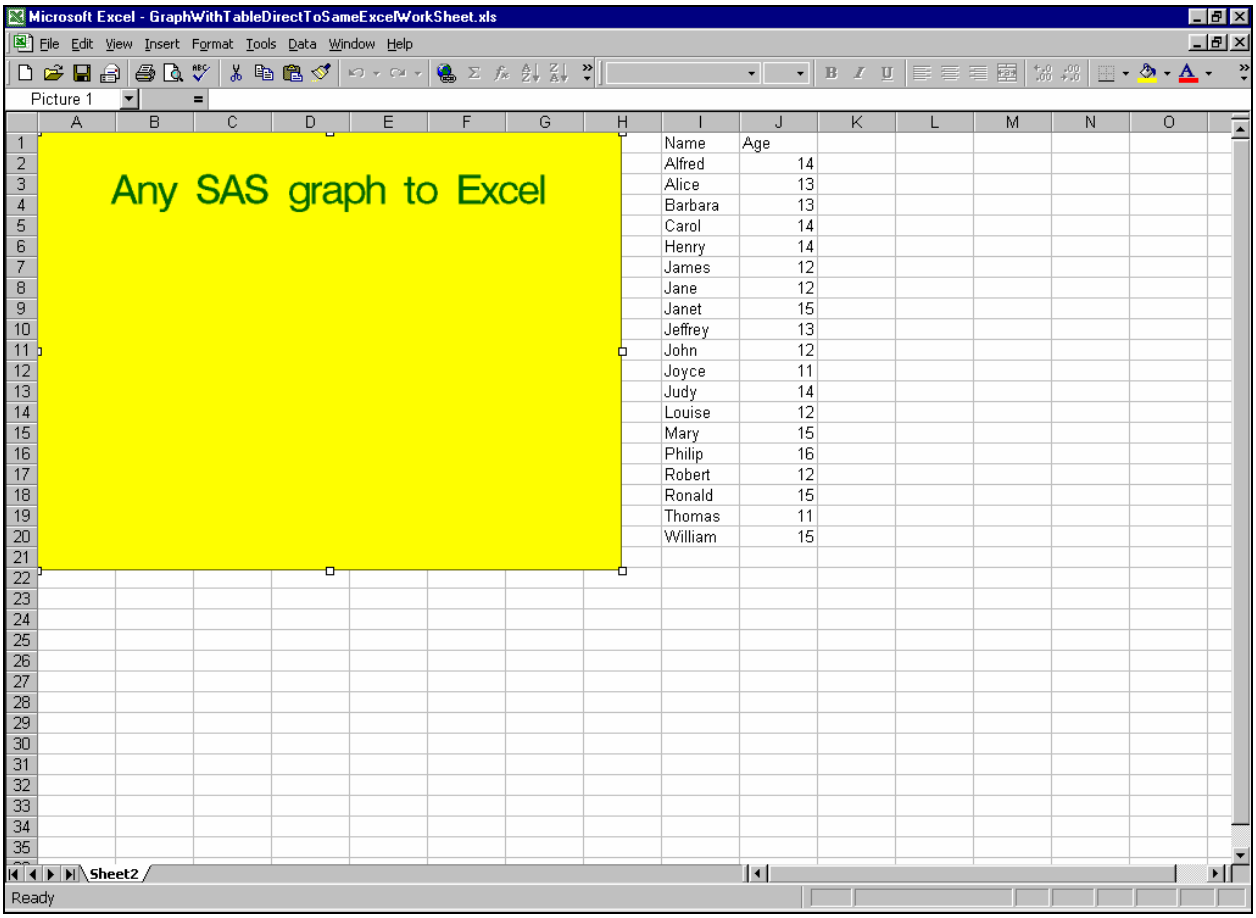
Four Graphs to the Same WorkSheet



Here is the code used to create the Excel worksheet above:

```
goptions reset=all;
goptions device=GIF;
goptions border;
goptions gsfmode=replace gsfname=anyname;
proc gslide;
filename anyname "C:\image1.GIF";
goptions cback=CXFFFF00;
title1 ' ' justify=CENTER height=4 color=CX006600
'UL SAS graph to Excel'
justify=CENTER
'OK Color Pair';
run;
filename anyname "C:\image2.GIF";
goptions cback=CXFF0000;
title1 ' ' justify=CENTER height=4 color=CX0000FF
'UR SAS graph to Excel'
justify=CENTER
'"Strong" Contrast';
run;
proc gslide;
filename anyname "C:\image3.GIF";
goptions cback=CXFF00FF; /* Use RGB Magenta for background. */
  /* Use RGB Cyan (Turquoise) for text. */
```

```
title1 ' ' justify=CENTER height=4 color=CX00FFFF
'LL SAS graph to Excel'
justify=CENTER
'Annoying Combination';
run;
filename anyname "C:\image4.GIF";
goptions cback=CX000000; /* Use RGB Black for background. */
  /* Use RGB White for text. */
title1 ' ' justify=CENTER height=4 color=CXFFFFFF
'LR SAS graph to Excel'
justify=CENTER
'Best Contrast';
run;
quit;

options noxwait noxsync;

x '"C:\Program Files\Microsoft Office\Office\EXCEL.exe"';

data _null_;
z=sleep(3);
run;

filename DDEcmds dde "excel|system";

data _null_;
file DDEcmds;
put '[workbook.activate("Sheet2")]';
put '[select("R1C1")]'; /* really the default position after activate */
put '[insert.picture("C:\image1.GIF")]';
put '[format.size(280,210)]';
put '[select("R1C7")]';
put '[insert.picture("C:\image2.GIF")]';
put '[format.size(280,210)]';
put '[select("R18C1")]';
put '[insert.picture("C:\image3.GIF")]';
put '[format.size(280,210)]';
put '[select("R18C7")]';
put '[insert.picture("C:\image4.GIF")]';
put '[format.size(280,210)]';
put '[error(false)]';
put '[workbook.delete("Sheet1")]';
put '[error(false)]';
put '[workbook.delete("Sheet3")]';
put '[error(false)]';
put '[save.as("C:\MultipleGraphsDirectToSameExcelWorkSheet.xls")]';
z=sleep(3);
put '[error(false)]';
put '[quit()]';
run;
```



Here is the code used to create the Excel worksheet above:

```
goptions reset=all;
goptions device=GIF;
goptions border;
goptions gsfmode=replace gsfname=anyname;
proc gslide;
filename anyname "C:\image1.GIF";
goptions cback=CXFFFF00;
title1 ' ' justify=CENTER height=4 color=CX006600 'Any SAS graph to Excel';
run; quit;

options noxwait noxsync;

 /* for your version of the Microsoft Office suite,
    you may need to use a different folder path below. */

x '"C:\Program Files\Microsoft Office\Office\EXCEL.exe"';

data _null_;
z=sleep(3);
run;

filename DDEcmds dde "excel|system";

data _null_;
file DDEcmds;
put '[workbook.activate("Sheet2")]';
```

```
put '[insert.picture("C:\image1.GIF")]';
put '[format.size(360,270)]';
run;

 /* insert labels for the table */

filename Labels DDE 'excel|Sheet2!r1c9:r1c10';

data _null_;
file Labels;
LABEL1 = 'Name';
LABEL2 = 'Age';
put LABEL1 LABEL2;
run;

 /* load the table data */

filename TblData DDE 'excel|Sheet2!r2c9:r20c10';

data _null_;
file TblData;
set sashelp.class;
put Name Age;
run;

 /* At this point, DDE could be used to customize the appearance of the table. */
 /* However, it will be left As Is. */

data _null_;
file DDEcmds;
put '[error(false)]';
put '[workbook.delete("Sheet1")]';
put '[error(false)]';
put '[workbook.delete("Sheet3")]';
put '[error(false)]';
put '[save.as("C:\GraphWithTableDirectToSameExcelWorkSheet.xls")]';
z=sleep(3);
put '[error(false)]';
put '[quit()]';
run;
```

Appendix 4: SAS Creation of Excel SubTotals

This information was not included in the original version of the paper. It includes: (a) text copied from the extra slides in the presentation; and (b) captioned slides of Excel screen print images.

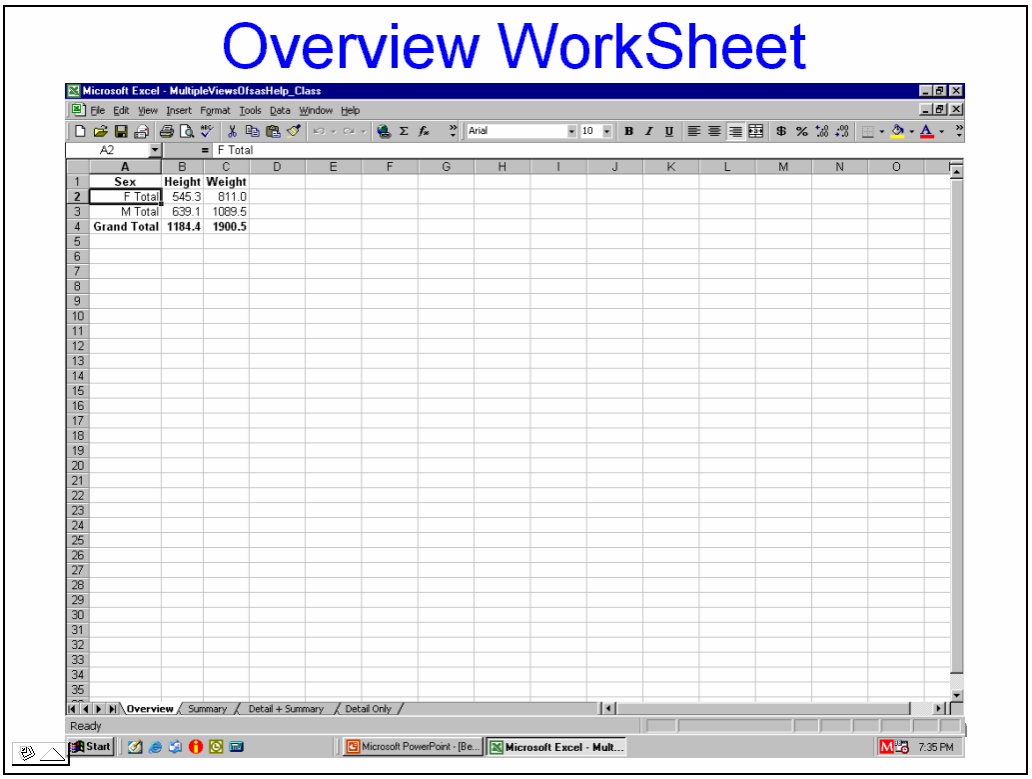
Excel Subtotals: Two Solutions

1. Separate Custom Worksheets for each level
2. DDE use of Excel's subtotals (**r**equires a pre-built Excel macro for one step)

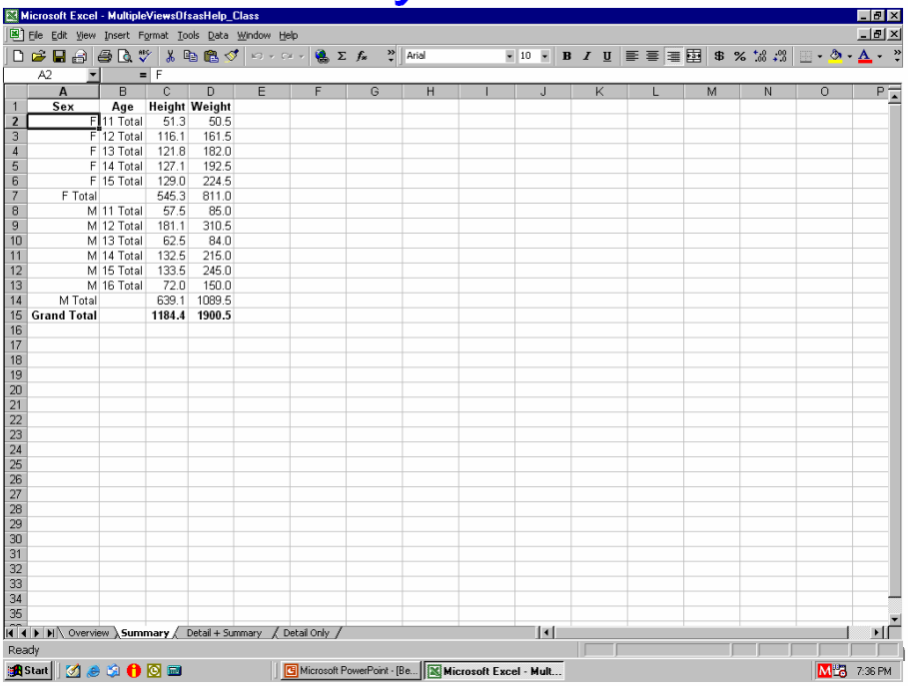
Excel Subtotals Solution 1: Separate Custom Worksheets for Each Level

An Alternative to Excel Subtotals

- Custom Design: Really "Have It Your Way"
- Prepare multiple pre-formatted empty WorkSheets
- Load them via DDE
- May have to use Software Intelligence for post-load formatting
- User moves from "level" to "level via Tab Navigation, not Excel "Level Buttons"
- Here showing only results, not code

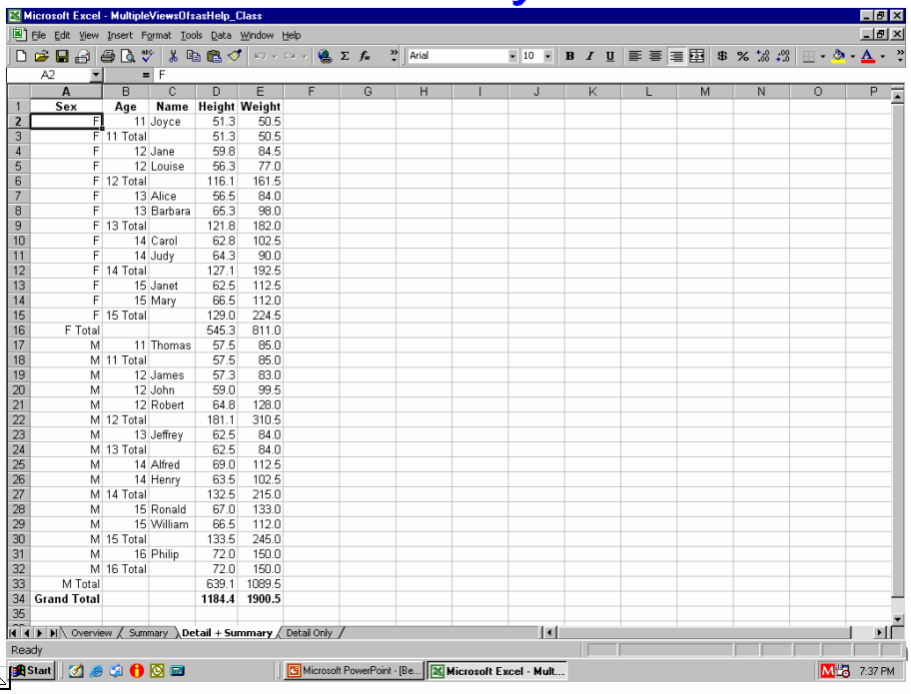


Summary WorkSheet

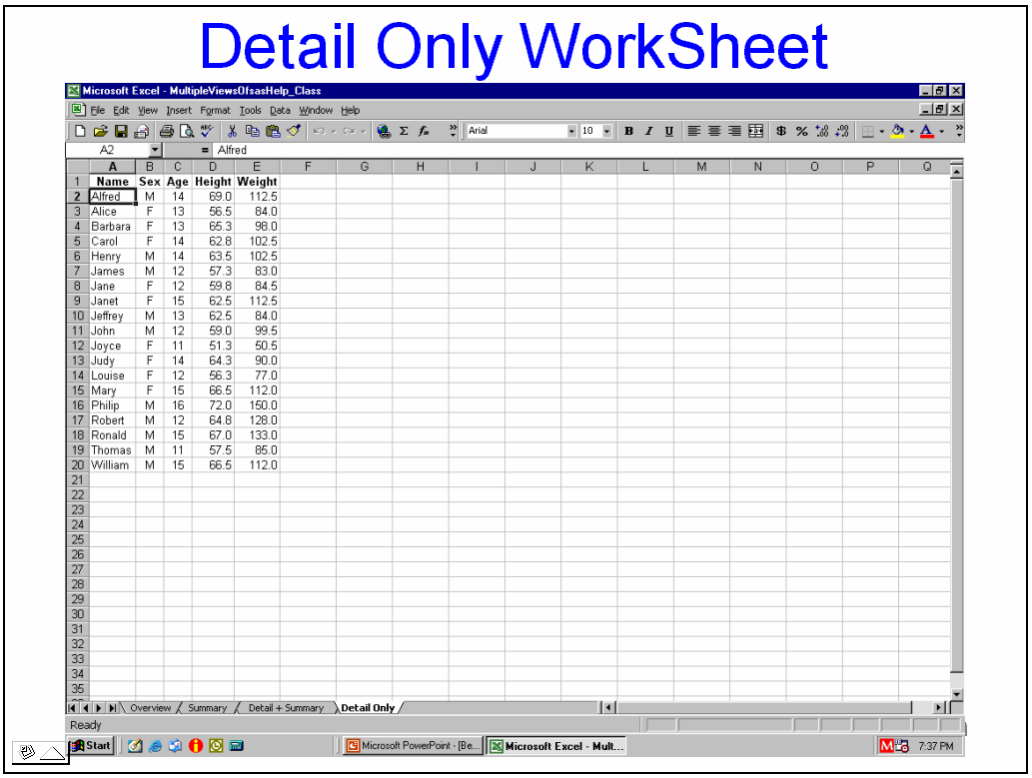


	Sex	Age	Height	Weight
2	F	11 Total	51.3	50.5
3	F	12 Total	116.1	161.5
4	F	13 Total	121.8	182.0
5	F	14 Total	127.1	192.5
6	F	15 Total	129.0	224.5
7	F Total		545.3	811.0
8	M	11 Total	57.5	85.0
9	M	12 Total	181.1	310.5
10	M	13 Total	62.5	84.0
11	M	14 Total	132.5	215.0
12	M	15 Total	133.5	245.0
13	M	16 Total	72.0	150.0
14	M Total		639.1	1089.5
15	**Grand Total**		**1184.4**	**1900.5**

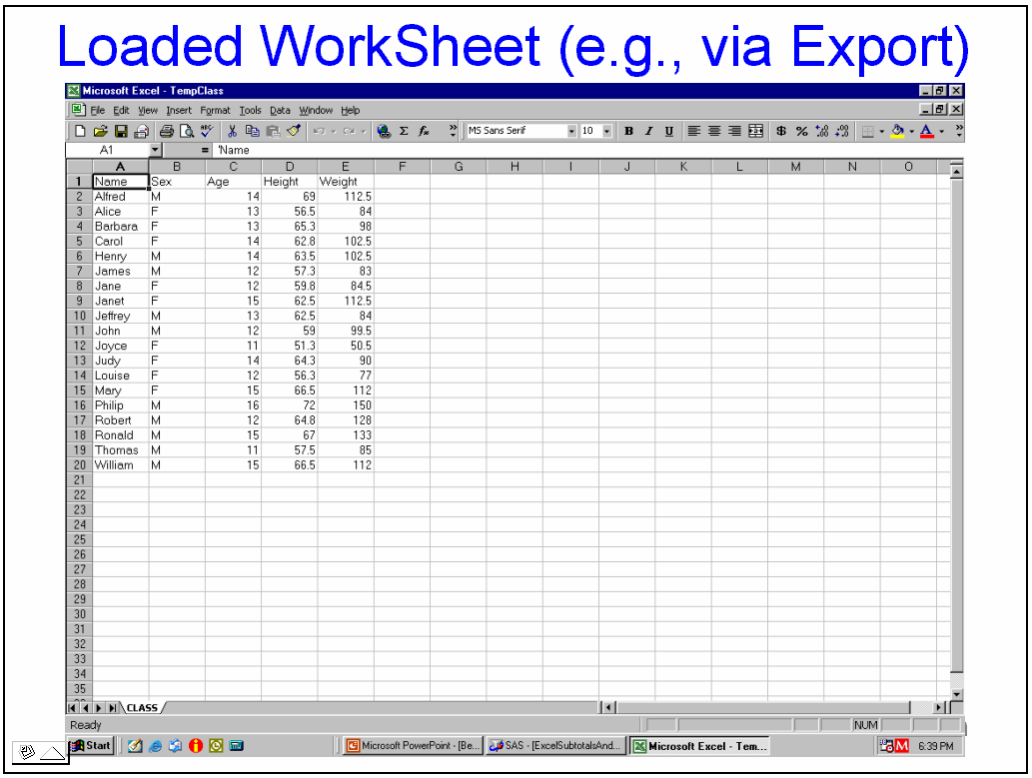
Detail + Summary WorkSheet



	Sex	Age	Name	Height	Weight
2	F	11	Joyce	51.3	50.5
3	F	11 Total		51.3	50.5
4	F	12	Jane	59.8	84.5
5	F	12	Louise	56.3	77.0
6	F	12 Total		116.1	161.5
7	F	13	Alice	56.5	84.0
8	F	13	Barbara	65.3	98.0
9	F	13 Total		121.8	182.0
10	F	14	Carol	62.8	102.5
11	F	14	Judy	64.3	90.0
12	F	14 Total		127.1	192.5
13	F	15	Janet	62.5	112.5
14	F	15	Mary	66.5	112.0
15	F	15 Total		129.0	224.5
16	F Total			545.3	811.0
17	M	11	Thomas	57.5	85.0
18	M	11 Total		57.5	85.0
19	M	12	James	57.3	83.0
20	M	12	John	59.0	99.5
21	M	12	Robert	64.8	128.0
22	M	12 Total		181.1	310.5
23	M	13	Jeffrey	62.5	84.0
24	M	13 Total		62.5	84.0
25	M	14	Alfred	69.0	112.5
26	M	14	Henry	63.5	102.5
27	M	14 Total		132.5	215.0
28	M	15	Ronald	67.0	133.0
29	M	15	William	66.5	112.0
30	M	15 Total		133.5	245.0
31	M	16	Philip	72.0	150.0
32	M	16 Total		72.0	150.0
33	M Total			639.1	1089.5
34	**Grand Total**			**1184.4**	**1900.5**

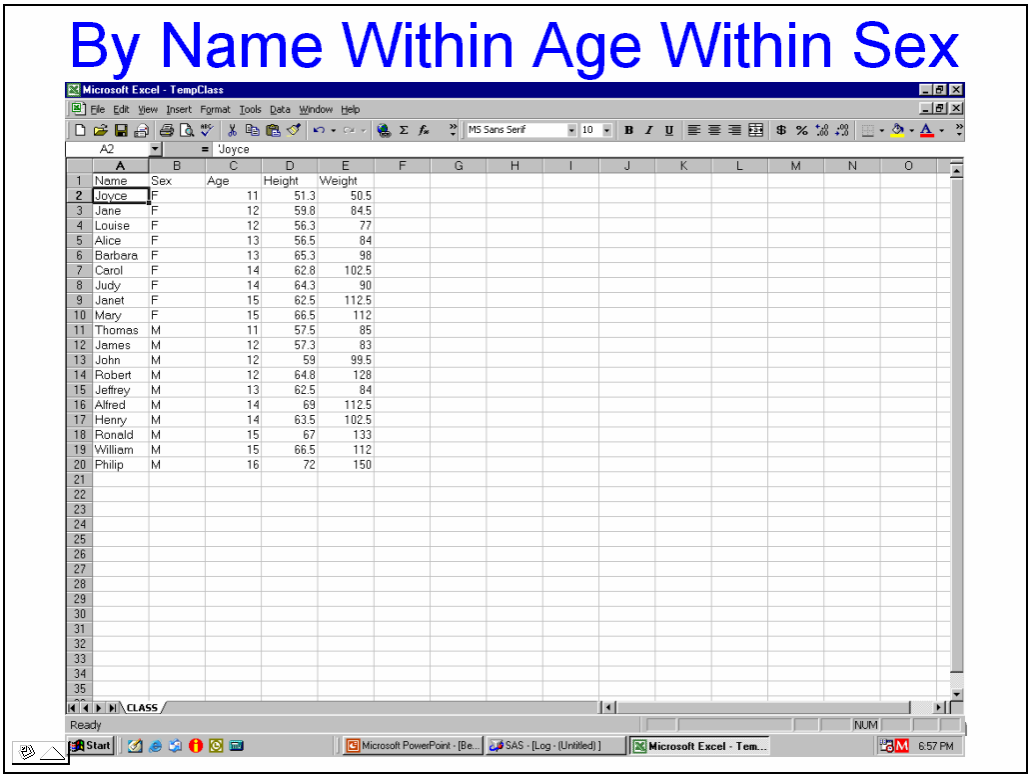


Excel Subtotals Solution 2: DDE Use of Excel's Subtotals



Sort the Data

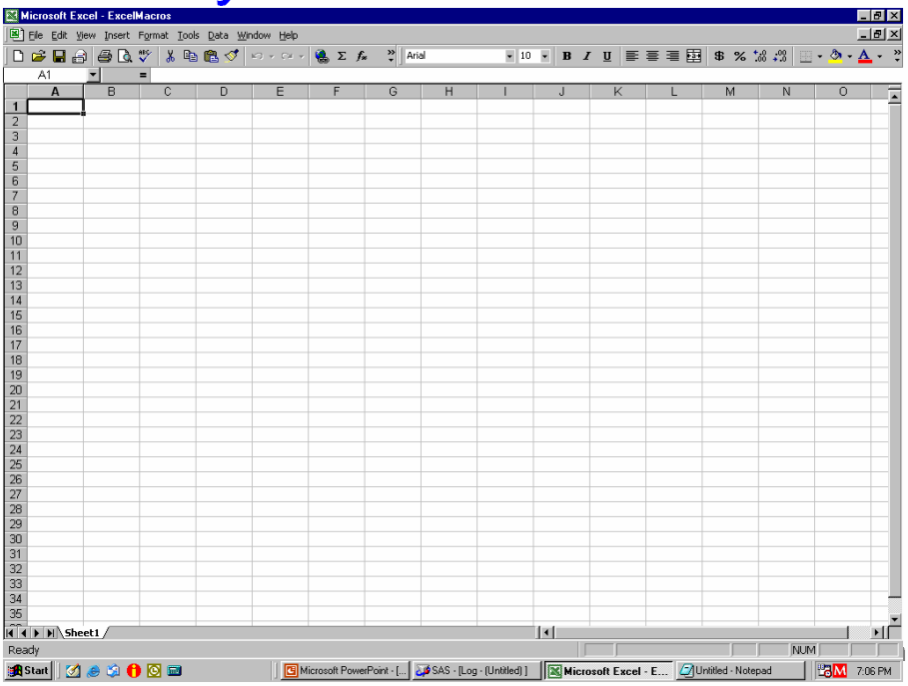
```
data _null_;
file DDEcmds;
put '[select("R1C1:R65536C256")]; /* all cells */
put '[sort(1,"Sex",1,"Age",1,"Name",1,1,1,0)]';
put '[select("R2C1")]';
run;
/* "Key",1 = sort on Key ascending */
/* 1 = sort by rows */
/* 1,1,0 = Normal sort, Headers present, Not case sensitive */
```



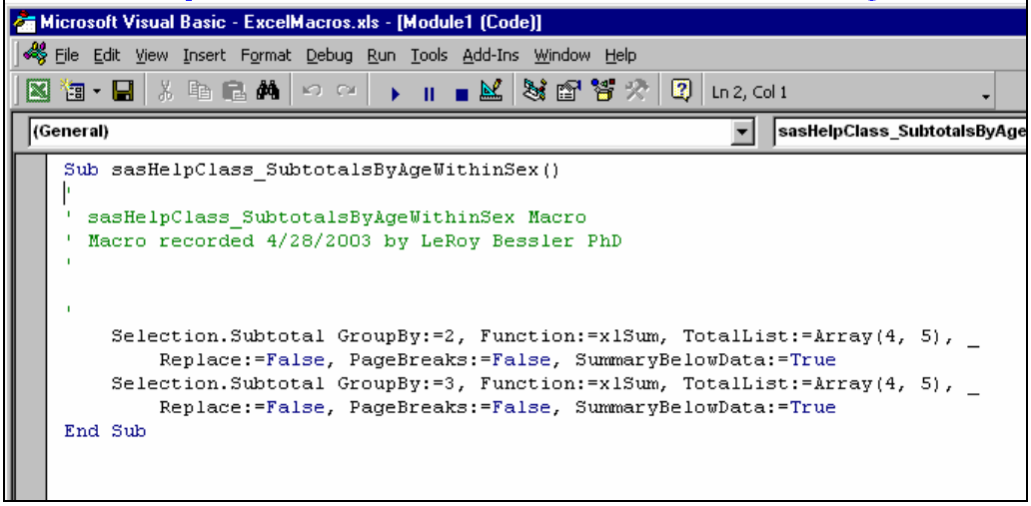
Close Data Workbook, Open a Workbook of Only Excel Macros

```
data _null_;
file DDEcmds;
put '[close(true)]';
x=sleep(1);
put '[open("C:\Folder\ExcelMacros.xls")]';
x=sleep(3);
run;
```

Only Excel Macros Here



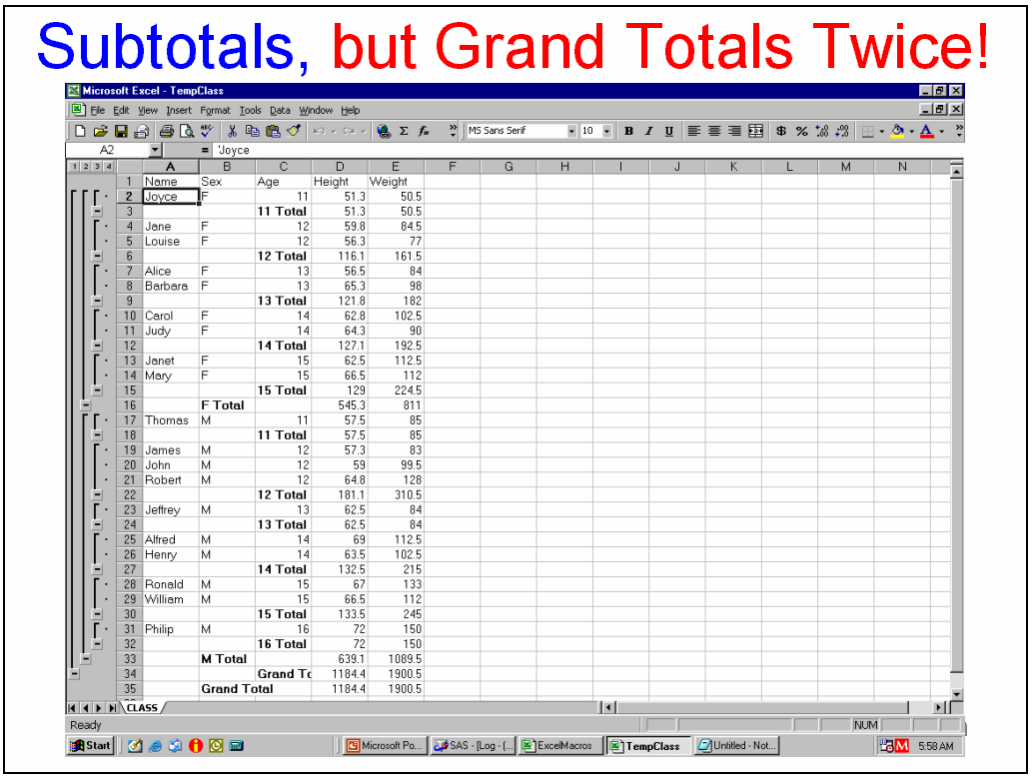
Open Macro for Demo Only

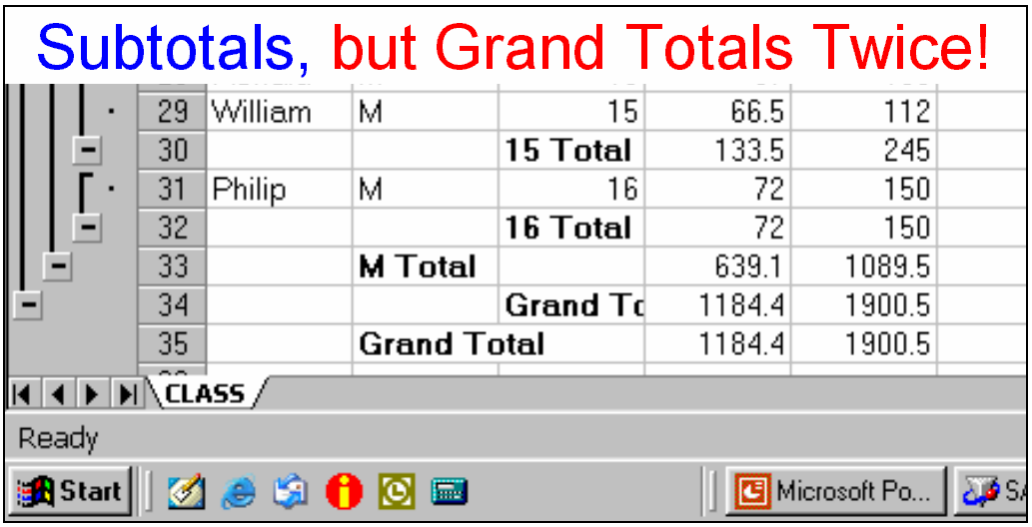



```
Sub sasHelpClass_SubtotalsByAgeWithinSex()
'
' sasHelpClass_SubtotalsByAgeWithinSex Macro
' Macro recorded 4/28/2003 by LeRoy Bessler PhD
'

'
    Selection.Subtotal GroupBy:=2, Function:=xlSum, TotalList:=Array(4, 5), _
        Replace:=False, PageBreaks:=False, SummaryBelowData:=True
    Selection.Subtotal GroupBy:=3, Function:=xlSum, TotalList:=Array(4, 5), _
        Replace:=False, PageBreaks:=False, SummaryBelowData:=True
End Sub
```

Open Data Workbook, and Run the Subtotals Excel Macro

```
data _null_;
file DDEcmds;
put '[open("C:\Folder\TempClass.xls")]';
x=sleep(3);
put '[run("ExcelMacros.xls!DoSubtotals",false)]';
run;
```





Removing the Extra Grand Total

First, Reformat & Save

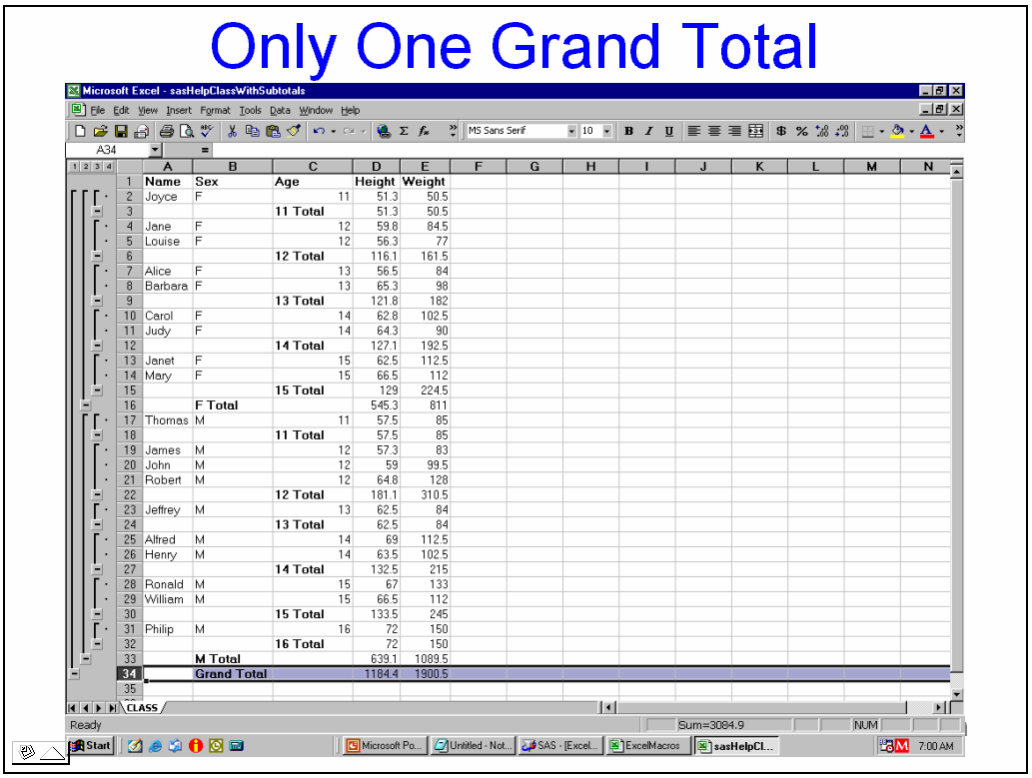
```
data _null_;
file DDEcmds;
put '[select("R1")]';
put '[font.properties("","Bold")]';
put '[select("C1:C5")]';
put '[column.width(,,,3)]';
put '[error(false)]';
put '[save.as("C:\Folder\DataWithSubtotals.xls")]';
x=sleep(1);
run;
```

Remove Extra Grand Total, Part 1:
Find Second from Last Row & Put Row Number in Symbol Table

```
filename Column DDE "EXCEL|CLASS!R1C1:R65536C1" notab;
data _null_;
length FirstCharInColumnCell $ 1;
infile Column truncover end=LastObs;
input FirstCharInColumnCell;
if LastObs;
call symput("RowToRemove",trim(left(_N_-1)));
run;
```

Remove Extra Grand Total, Part 2:
Select Second from Last Row & Delete It

```
data _null_; file DDEcmds;
put '[select("R' "&RowToRemove" '")]';
/* resolves to: put '[select("R34")]'; */
put '[edit.delete(3)]';
run;
```



Changing & Indenting Values
and
Changing Excel Subtotal Labels

Pseudo-Indent Cell Content

```
data _null_;
file DDEcmds;
put '[select("R2C2:R15C2")]';
put   '[formula.replace("F","___Female",1,,false,false)]';

/* change color of three _ to White to match cell background: */
put '[font.properties(,,,,,,,,2,,,1,3)]';
     /* 2 = White , StartAt = 1, ForLengthOf = 3 */

run;
```

Indent Work In Progress

Microsoft Excel - sasHelpClassWithSubtotals

File Edit View Insert Format Tools Data Window Help

B2 = ' Female

	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Joyce	Female	11	51.3	50.5
3			11 Total	51.3	50.5
4	Jane	Female	12	59.8	84.5
5	Louise	Female	12	56.3	77
6			12 Total	116.1	161.5
7	Alice	Female	13	56.5	84
8	Barbara	Female	13	65.3	98
9			13 Total	121.8	182
10	Carol	Female	14	62.8	102.5
11	Judy	Female	14	64.3	90
12			14 Total	127.1	192.5
13	Janet	Female	15	62.5	112.5
14	Mary	Female	15	66.5	112
15			15 Total	129	224.5
16		F Total		545.3	811
17	Thomas	M	11	57.5	85
18			11 Total	57.5	85
19	James	M	12	57.3	83
20	John	M	12	59	99.5
21	Robert	M	12	64.8	128
22			12 Total	181.1	310.5
23	Jeffrey	M	13	62.5	84
24			13 Total	62.5	84

Indents Complete

Microsoft Excel - sasHelpClassWithSubtotals

File Edit View Insert Format Tools Data Window Help

A1 = 'Name

	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Joyce	Female	11	51.3	50.5
3			11 Total	51.3	50.5
4	Jane	Female	12	59.8	84.5
5	Louise	Female	12	56.3	77
6			12 Total	116.1	161.5
7	Alice	Female	13	56.5	84
8	Barbara	Female	13	65.3	98
9			13 Total	121.8	182
10	Carol	Female	14	62.8	102.5
11	Judy	Female	14	64.3	90
12			14 Total	127.1	192.5
13	Janet	Female	15	62.5	112.5
14	Mary	Female	15	66.5	112
15			15 Total	129	224.5
16		F Total		545.3	811
17	Thomas	Male	11	57.5	85
18			11 Total	57.5	85
19	James	Male	12	57.3	83
20	John	Male	12	59	99.5
21	Robert	Male	12	64.8	128
22			12 Total	181.1	310.5
23	Jeffrey	Male	13	62.5	84
24			13 Total	62.5	84

New Labels for Sex Subtotals

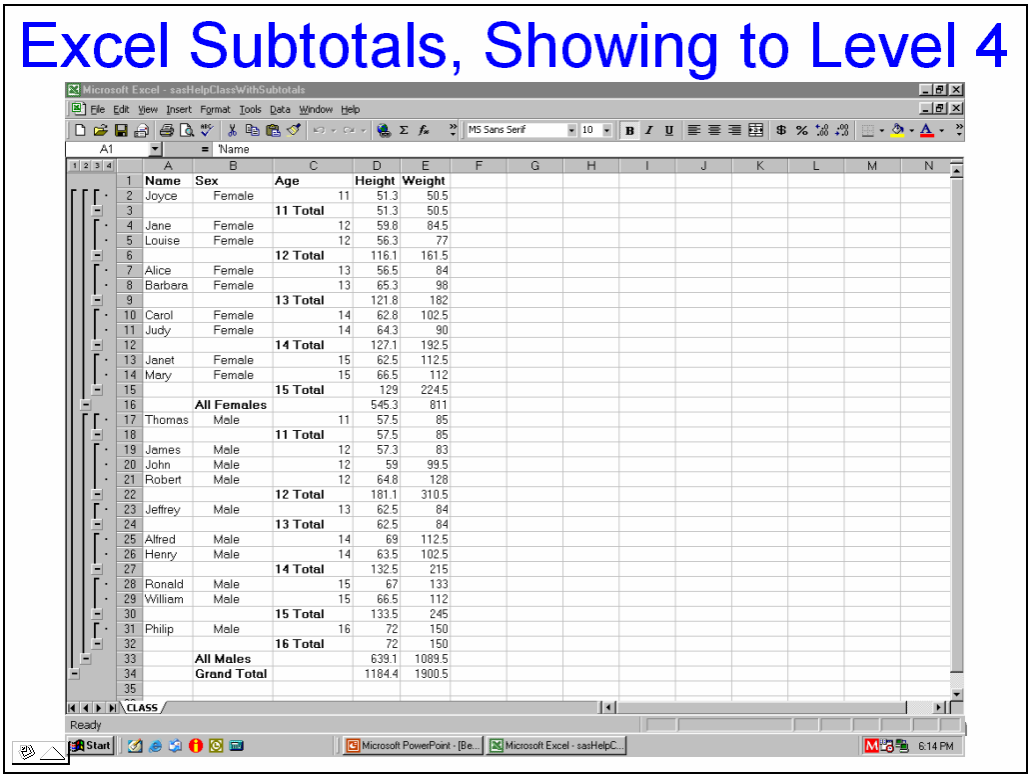
Microsoft Excel - sasHelpClassWithSubtotals

File Edit View Insert Format Tools Data Window Help

A1 = 'Name

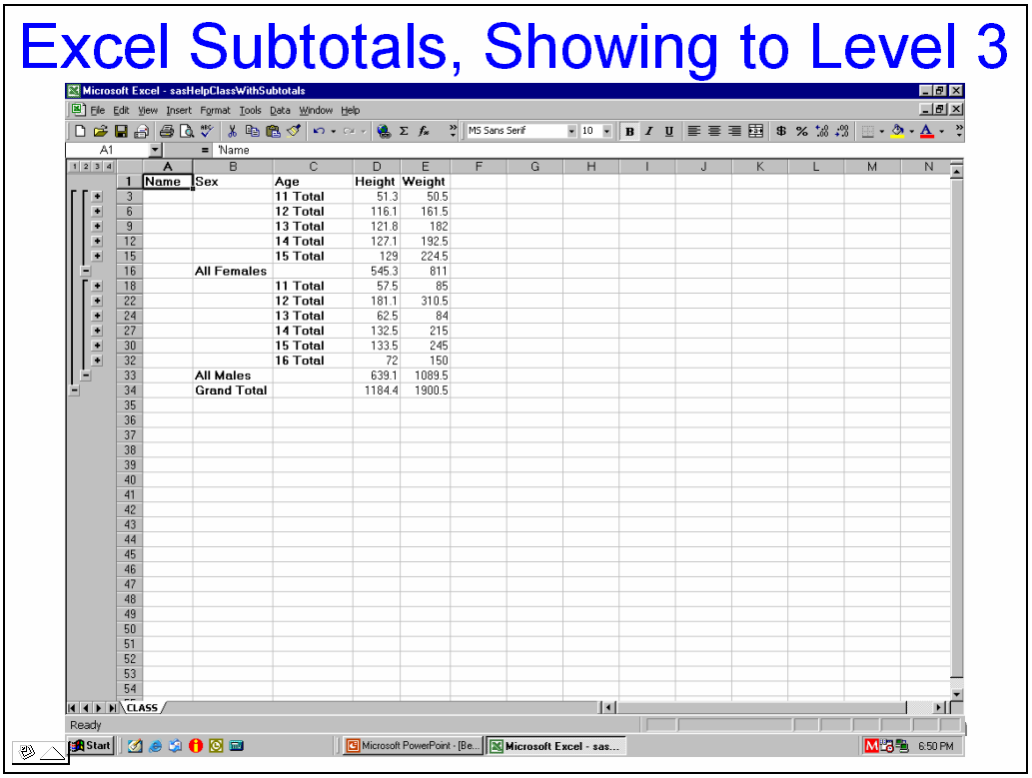
	A	B	C	D	E
1	Name	Sex	Age	Height	Weight
2	Joyce	Female	11	51.3	50.5
3			11 Total	51.3	50.5
4	Jane	Female	12	59.8	84.5
5	Louise	Female	12	56.3	77
6			12 Total	116.1	161.5
7	Alice	Female	13	56.5	84
8	Barbara	Female	13	65.3	98
9			13 Total	121.8	182
10	Carol	Female	14	62.8	102.5
11	Judy	Female	14	64.3	90
12			14 Total	127.1	192.5
13	Janet	Female	15	62.5	112.5
14	Mary	Female	15	66.5	112
15			15 Total	129	224.5
16		All Females		545.3	811
17	Thomas	Male	11	57.5	85
18			11 Total	57.5	85
19	James	Male	12	57.3	83
20	John	Male	12	59	99.5
21	Robert	Male	12	64.8	128
22			12 Total	181.1	310.5
23	Jeffrey	Male	13	62.5	84
24			13 Total	62.5	84

Controlling Subtotal Display

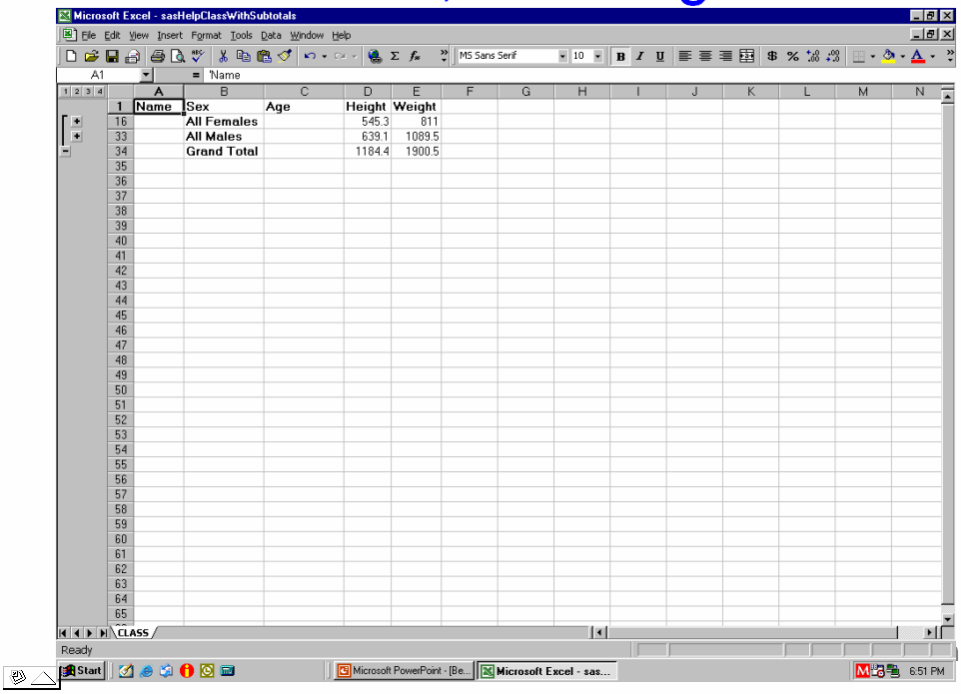


Show Subtotals Only to Level 3:

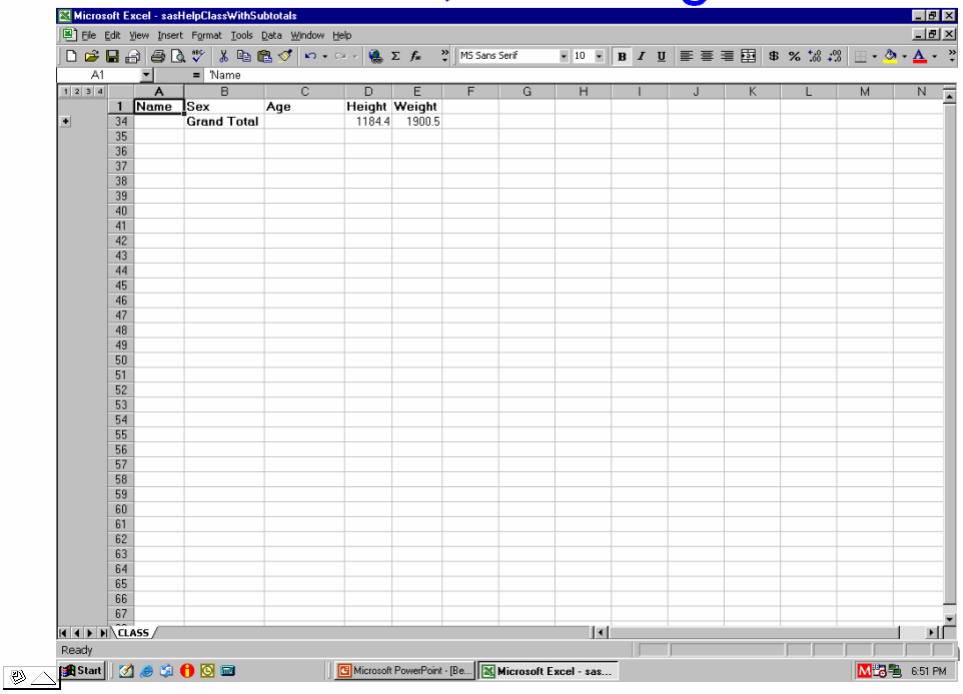
```
data _null_; file DDEcmds; put '[show.levels(3,0)]'; run;
```



Excel Subtotals, Showing to Level 2



Excel Subtotals, Showing to Level 1



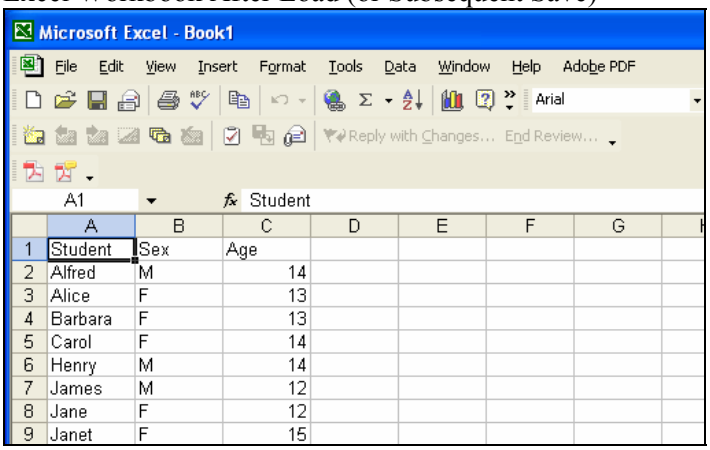
Appendix 5: SAS DDE to Excel 2007 and PDF

NOTE: This information was not included in the original version of the paper, nor was it included in the expanded edition of that paper, which included Appendices 1-4 above.

The method used here was discovered by Craig Wildeman, by trial and error, since Microsoft does not document the file type values needed in Save.As with DDE. Reach him at Craig.Wildeman@kohler.com

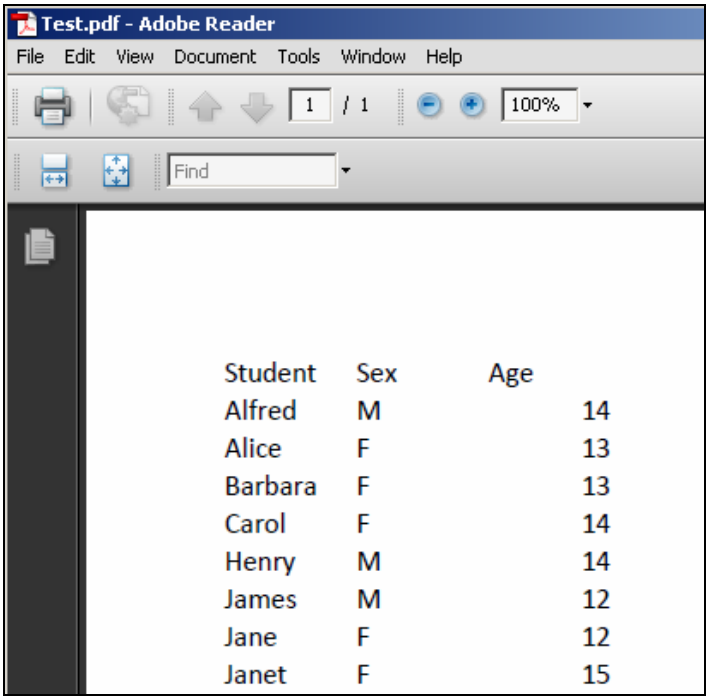
Typically one would create a PDF directly from SAS using ODS PDF. However, if you need to deliver the same information, possibly to different recipients, in a comparable layout as an Excel 2007 spreadsheet or a matching PDF, there is a Dynamic Data Exchange (DDE) solution. The spreadsheet data can be used however recipients wish, and the PDF is read-only unalterable information for recipients.

Excel Workbook After Load (or Subsequent Save)



	A	B	C	D	E	F	G
1	Student	Sex	Age				
2	Alfred	M	14				
3	Alice	F	13				
4	Barbara	F	13				
5	Carol	F	14				
6	Henry	M	14				
7	James	M	12				
8	Jane	F	12				
9	Janet	F	15				

Saved As PDF



Here is the code used to create the PDF above:

```
OPTIONS NOXWAIT NOXSYNC;

X '"C:\Program Files\Microsoft Office\Office12\EXCEL.exe"';
* Start Excel 2007 *;
* The later DDE save as PDF only works with Excel 2007 *;

DATA _NULL_;
x=SLEEP(3); /* Wait 3 seconds for EXCEL to start */
RUN;

FILENAME DDEcmds DDE "excel|system";

FILENAME hdgcol1 DDE 'Excel|Sheet1!R1C1';
FILENAME hdgcol2 DDE 'Excel|Sheet1!R1C2';
FILENAME hdgcol3 DDE 'Excel|Sheet1!R1C3';
FILENAME TheData DDE 'Excel|Sheet1!R2C1:R20C3';

DATA _NULL_;
FILE hdgcol1; PUT 'Student';
FILE hdgcol2; PUT 'Sex';
FILE hdgcol3; PUT 'Age';
set sashelp.class;
FILE TheData;
PUT name sex age;
RUN;

FILENAME hdgcol1 CLEAR;
FILENAME hdgcol2 CLEAR;
FILENAME hdgcol3 CLEAR;
FILENAME TheData CLEAR;

DATA _NULL_;
FILE DDEcmds;
PUT '[error(false)]'; /* Suppress Excel prompts to verify your
intentions. E.g., whether or not you want to replace a pre-existing
version of this file. */
PUT '[save.as("C:\DDEoutput\Test",1)]';
RUN;

 /* To save the default three-sheet workbook as PDF, TXT, or CSV
without needing to reply to prompts, delete the empty worksheets. */

DATA _NULL_;
FILE DDEcmds;
PUT '[workbook.delete("Sheet2")]';
PUT '[workbook.delete("Sheet3")]';
PUT '[error(false)]';
PUT '[save.as("C:\DDEoutput\Test",57)]';
RUN;
 /* 57 is the file type for PDF. This only works with Excel 2007. */
```

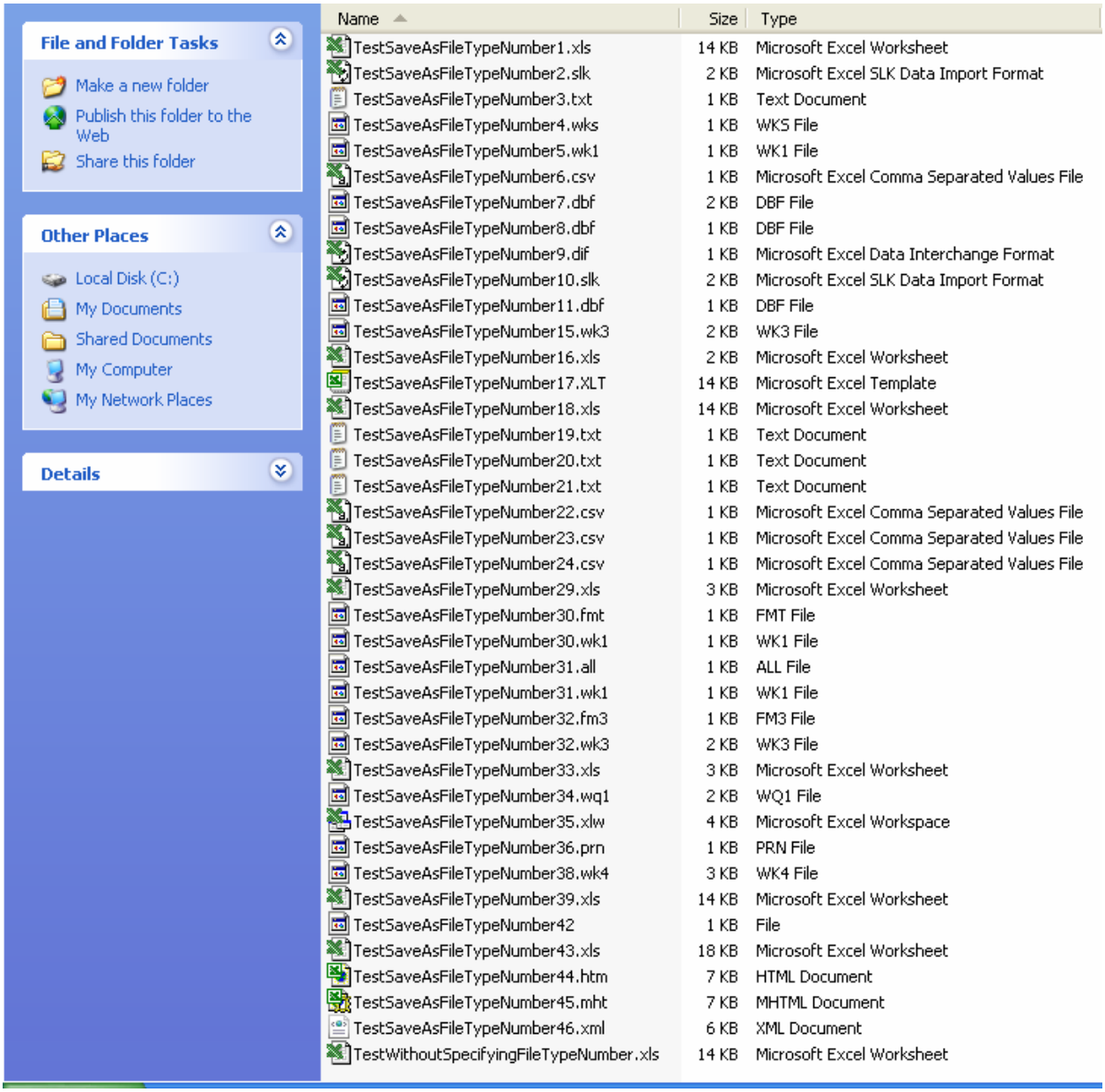
```
DATA _NULL_;
FILE DDEcmds;
PUT '[error(false)]';
PUT '[quit()]';
RUN;

FILENAME DDEcmds CLEAR;
```

Appendix 6: SAS DDE with Excel 2002 & Excel 2007, Saving to Other File Types

This is a extension of the work in Appendix 5 to examine what other file types can be saved besides PDF. PDF is available as an option if you are working with Excel 2007. The results in this Appendix were obtained by using Excel 2002 and Excel 2007 as the Excel server for Version 9.2 of SAS. The Excel 2007 results were provided by Craig Wildeman.

Presented are the file types created with Excel 2002, the file types created with Excel 2007, the code used, an excerpt from a SAS log, and the only documentation that I could find to: (a) explain the difference between file type numbers with the same file type description; and (b) provide a list of saveable file types.

Here is a list of file types created from Excel 2002 trying every file type number in the range 1 to 100:



Name	Size	Type
TestSaveAsFileTypeNumber1.xls	14 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber2.slk	2 KB	Microsoft Excel SLK Data Import Format
TestSaveAsFileTypeNumber3.txt	1 KB	Text Document
TestSaveAsFileTypeNumber4.wks	1 KB	WKS File
TestSaveAsFileTypeNumber5.wk1	1 KB	WK1 File
TestSaveAsFileTypeNumber6.csv	1 KB	Microsoft Excel Comma Separated Values File
TestSaveAsFileTypeNumber7.dbf	2 KB	DBF File
TestSaveAsFileTypeNumber8.dbf	1 KB	DBF File
TestSaveAsFileTypeNumber9.dif	1 KB	Microsoft Excel Data Interchange Format
TestSaveAsFileTypeNumber10.slk	2 KB	Microsoft Excel SLK Data Import Format
TestSaveAsFileTypeNumber11.dbf	1 KB	DBF File
TestSaveAsFileTypeNumber15.wk3	2 KB	WK3 File
TestSaveAsFileTypeNumber16.xls	2 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber17.XLT	14 KB	Microsoft Excel Template
TestSaveAsFileTypeNumber18.xls	14 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber19.txt	1 KB	Text Document
TestSaveAsFileTypeNumber20.txt	1 KB	Text Document
TestSaveAsFileTypeNumber21.txt	1 KB	Text Document
TestSaveAsFileTypeNumber22.csv	1 KB	Microsoft Excel Comma Separated Values File
TestSaveAsFileTypeNumber23.csv	1 KB	Microsoft Excel Comma Separated Values File
TestSaveAsFileTypeNumber24.csv	1 KB	Microsoft Excel Comma Separated Values File
TestSaveAsFileTypeNumber29.xls	3 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber30.fmt	1 KB	FMT File
TestSaveAsFileTypeNumber30.wk1	1 KB	WK1 File
TestSaveAsFileTypeNumber31.all	1 KB	ALL File
TestSaveAsFileTypeNumber31.wk1	1 KB	WK1 File
TestSaveAsFileTypeNumber32.fm3	1 KB	FM3 File
TestSaveAsFileTypeNumber32.wk3	2 KB	WK3 File
TestSaveAsFileTypeNumber33.xls	3 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber34.wq1	2 KB	WQ1 File
TestSaveAsFileTypeNumber35.xlw	4 KB	Microsoft Excel Workspace
TestSaveAsFileTypeNumber36.prn	1 KB	PRN File
TestSaveAsFileTypeNumber38.wk4	3 KB	WK4 File
TestSaveAsFileTypeNumber39.xls	14 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber42	1 KB	File
TestSaveAsFileTypeNumber43.xls	18 KB	Microsoft Excel Worksheet
TestSaveAsFileTypeNumber44.htm	7 KB	HTML Document
TestSaveAsFileTypeNumber45.mht	7 KB	MHTML Document
TestSaveAsFileTypeNumber46.xml	6 KB	XML Document
TestWithoutSpecifyingFileTypeNumber.xls	14 KB	Microsoft Excel Worksheet

Below is a list of file types created from Excel 2007 trying every file type number in the range 1 to 100. Every file type number in the range 50 to 60, except for 59, now works, but many file types supported by Excel 2002 are not supported from Excel 2007.

Name	Size	Type
TestSaveAsFileTypeNumber1.xls	17 KB	Microsoft Office Excel 97-2003 Worksheet
TestSaveAsFileTypeNumber2.slk	3 KB	Microsoft Office Excel SLK Data Import Format
TestSaveAsFileTypeNumber3.txt	1 KB	Text Document
TestSaveAsFileTypeNumber6.csv	1 KB	Microsoft Office Excel Comma Separated Values File
TestSaveAsFileTypeNumber9.dif	1 KB	DIF File
TestSaveAsFileTypeNumber10.slk	3 KB	Microsoft Office Excel SLK Data Import Format
TestSaveAsFileTypeNumber17.xlt	17 KB	Microsoft Office Excel Template
TestSaveAsFileTypeNumber18.xls	17 KB	Microsoft Office Excel 97-2003 Worksheet
TestSaveAsFileTypeNumber19.txt	1 KB	Text Document
TestSaveAsFileTypeNumber20.txt	1 KB	Text Document
TestSaveAsFileTypeNumber21.txt	1 KB	Text Document
TestSaveAsFileTypeNumber22.csv	1 KB	Microsoft Office Excel Comma Separated Values File
TestSaveAsFileTypeNumber23.csv	1 KB	Microsoft Office Excel Comma Separated Values File
TestSaveAsFileTypeNumber24.csv	1 KB	Microsoft Office Excel Comma Separated Values File
TestSaveAsFileTypeNumber36.prn	1 KB	PRN File
TestSaveAsFileTypeNumber39.xls	12 KB	Microsoft Office Excel 97-2003 Worksheet
TestSaveAsFileTypeNumber42	1 KB	File
TestSaveAsFileTypeNumber44.htm	10 KB	HTML Document
TestSaveAsFileTypeNumber45.mht	19 KB	MHTML Document
TestSaveAsFileTypeNumber46.xml	6 KB	XML Document
TestSaveAsFileTypeNumber50.xlsb	8 KB	Microsoft Office Excel Binary Worksheet
TestSaveAsFileTypeNumber51.xlsx	8 KB	Microsoft Office Excel Worksheet
TestSaveAsFileTypeNumber52.xlsm	8 KB	Microsoft Office Excel Macro-Enabled Worksheet
TestSaveAsFileTypeNumber53.xltm	8 KB	Microsoft Office Excel Macro-Enabled Template
TestSaveAsFileTypeNumber54.xltx	8 KB	Microsoft Office Excel Template
TestSaveAsFileTypeNumber55.xlam	8 KB	Microsoft Office Excel Add-In
TestSaveAsFileTypeNumber56.xls	18 KB	Microsoft Office Excel 97-2003 Worksheet
TestSaveAsFileTypeNumber57.pdf	69 KB	Adobe Acrobat Document
TestSaveAsFileTypeNumber58.xps	75 KB	XPS File
TestSaveAsFileTypeNumber60.ods	4 KB	OpenDocument Spreadsheet

Here is the code used for the exercises reported above:

```
OPTIONS NOXWAIT NOXSYNC;

X '"C:\Program Files\Microsoft Office\Office10\EXCEL.exe"';

/* Would need to use Microsoft Office\Office12 for Excel 2007 */

* The above command starts Excel *;

DATA _NULL_;
x=SLEEP(3); /* Wait 3 seconds for EXCEL to start */
RUN;

FILENAME DDEcmds DDE "excel|system";

FILENAME hdgcol1 DDE 'Excel|Sheet1!R1C1';
FILENAME hdgcol2 DDE 'Excel|Sheet1!R1C2';
FILENAME hdgcol3 DDE 'Excel|Sheet1!R1C3';
FILENAME TheData DDE 'Excel|Sheet1!R2C1:R20C3';

DATA _NULL_;
FILE hdgcol1; PUT 'Student';
FILE hdgcol2; PUT 'Sex';
FILE hdgcol3; PUT 'Age';
set sashelp.class;
FILE TheData;
PUT name sex age;
RUN;

FILENAME hdgcol1 CLEAR;
FILENAME hdgcol2 CLEAR;
FILENAME hdgcol3 CLEAR;
FILENAME TheData CLEAR;

DATA _NULL_;
FILE DDEcmds;
PUT '[error(false)]'; /* Suppress Excel prompts to verify your
intentions. E.g., whether or not you want to replace a pre-existing
version of this file. */
PUT '[save.as("C:\DDEoutput\TestWithoutSpecifyingFileTypeNumber")]';
RUN;

DATA _NULL_;
x=SLEEP(2); /* Wait 2 seconds for first Test to finish */
RUN;

DATA _NULL_;
FILE DDEcmds;
 /* To save the default three-sheet workbook as other file types
without needing to reply to prompts, delete the empty worksheets. */
```

```
PUT '[workbook.delete("Sheet2")]';
PUT '[workbook.delete("Sheet3")]';
RUN;

%macro SaveOtherFileTypes(From=,To=);

%do i = &From %to &To %by 1;

DATA _NULL_;
x=SLEEP(1); /* Wait 1 seconds for prior Save As to finish */
RUN;

%put ***************************;
%put Trying to Save.As File Type &i;

DATA _NULL_;
FILE DDEcmds;
PUT '[error(false)]'; /* Suppress Excel prompts to verify your
intentions. E.g., whether or not you want to replace a pre-existing
version of this file. */
PUT
%unquote(%str(%'[save.as("C:\DDEoutput\TestSaveAsFileTypeNumber&i",&i)
]%'));
RUN;

%end;

%mend SaveOtherFileTypes;

%SaveOtherFileTypes(From=1,To=100);

DATA _NULL_;
FILE DDEcmds;
PUT '[error(false)]';
PUT '[quit()]';
RUN;

FILENAME DDEcmds CLEAR;
```

Below is what you find in the SAS log for those file types that are not supported by the version of Excel that you are working with. For this test, Excel 2002 was being used, rather than Excel 2007.

```
****************************
Trying to Save.As File Type 57

NOTE: The file DDECMDS is:
      DDE Session,
      SESSION=excel|system,RECFM=V,LRECL=256

ERROR: DDE session not ready.
FATAL: Unrecoverable I/O error detected in the execution of the DATA step program.
       Aborted during the EXECUTION phase.
NOTE: 1 record was written to the file DDECMDS.
      The minimum record length was 14.
      The maximum record length was 14.
NOTE: The SAS System stopped processing this step because of errors.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

From macrofun.hlp for **Save.As**

Type_num	File format
1 or omitted	Normal
2	SYLK
3	Text
4	WKS
5	WK1
6	CSV
7	DBF2
8	DBF3
9	DIF
10	Reserved
11	DBF4
12	Reserved
13	Reserved
14	Reserved
15	WK3
16	Microsoft Excel 2.x
17	Template
18	Add-in macro (For compatibility only. In Microsoft Excel 5.0, this saves as normal.)
19	Text (Macintosh)
20	Text (Windows)
21	Text (MS-DOS)
22	CSV (Macintosh)
23	CSV (Windows)
24	CSV (MS-DOS)
25	International macro
26	International add-in macro
27	Reserved
28	Reserved
29	Microsoft Excel 3.0
30	WK1 / FMT
31	WK1 / Allways
32	WK3 / FM3
33	Microsoft Excel 4.0
34	WQ1
35	Microsoft Excel 4.0 workbook
36	Formatted text (space delimited)

From http://www.pcreview.co.uk/forums/thread-3779204.php

This is a dialogue between two people whose names are in the web page, but omitted here.

Question:

Is there a list of file formats and the numeric equivalent for the different Excel extensions available in 2007 (and 2003)? I'm using this in a SaveAs command and want to make sure I've covered my bases.

Answer:

Following in all versions since 2000:

xlAddIn 18
xlCSV 6
xlCSVMac 22
xlCSVMSDOS 24
xlCSVWindows 23
xlDBF2 7
xlDBF3 8
xlDBF4 11
xlDIF 9
xlExcel2 16
xlExcel2FarEast 27
xlExcel3 29
xlExcel4 33
xlExcel5 39
xlExcel7 39
xlExcel9795 43
xlExcel4Workbook 35
xlIntlAddIn 26
xlIntlMacro 25
xlWorkbookNormal -4143
xlSYLK 2
xlTemplate 17
xlCurrentPlatformText -4158
xlTextMac 19
xlTextMSDOS 21
xlTextPrinter 36
xlTextWindows 20
xlWJ2WD1 14
xlWK1 5
xlWK1ALL 31
xlWK1FMT 30
xlWK3 15
xlWK4 38
xlWK3FM3 32
xlWKS 4
xlWorks2FarEast 28
xlWQ1 34
xlWJ3 40

xlWJ3FJ3 41
xlUnicodeText 42
xlHtml 44

Following n/a in 2000, new in 2002 or 2003, not sure which:

xlWebArchive 45
xlXMLSpreadsheet 46

Following are new in 2007:

xlExcel12 50
xlOpenXMLWorkbook 51
xlOpenXMLWorkbookMacroEnabled 52
xlOpenXMLTemplateMacroEnabled 53
xlTemplate8 17
xlOpenXMLTemplate 54
xlAddIn8 18
xlOpenXMLAddIn 55
xlExcel8 56
xlWorkbookDefault 51